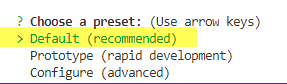
02\_setting\_up\_vuetify

第一步： create a vue project using vue.js

vue create todo-ninja

第二步: 将vuetify作为plugin 加入进来

vue add vuetify





03\_vuetify\_basics

1. 所有vuetify 组件一定要nested inside v-app, 因为它是mouting point for other 组件



1. Toolbar

[Toolbar](https://vuetifyjs.com/en/components/toolbars" \l "toolbar)

The **v-toolbar** component is pivotal to any gui, as it generally is the primary source of site navigation. The toolbar component works great in conjunction with a navigation drawer for hiding links and presenting an activator to open the sidebar on mobile.

如果你在浏览器观察，toolbar 实际上是由nav 和 div组成



几个常见的property:

1. class=”indigo” 背景色为蓝色
2. dark dark-theme, 默认为ligth
3. height=”70”
4. app 使得navbar fixed



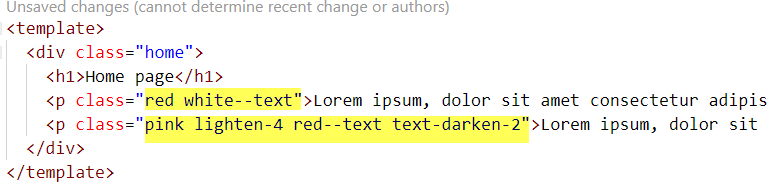


v-content

所有page的内容都应该在v-content内部



04\_text\_colours



背景的话直接用颜色代表，比如red, pink, 加深/减弱 也是直接写 lighten-4 或者darken-2

字的话用<color>--text, 比如white—text, 加深/减弱前面必须加上text-, 比如text-darken-2

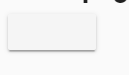
<https://vuetifyjs.com/en/framework/colors>

字体大小：

<https://vuetifyjs.com/en/framework/typography>



05\_button\_icons

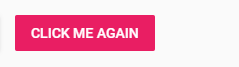
1. <v-btn></v-btn> 没有字的button  
   
2. <v-btn class="pink white--text">Click me</v-btn>

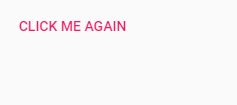


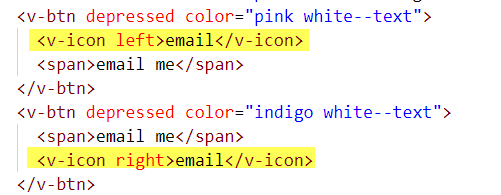
1. <v-btn dark color="pink">Click me again</v-btn>

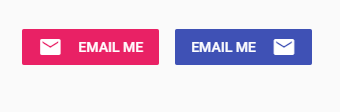


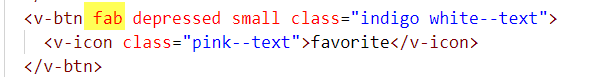
1. <v-btn depressed dark color="pink">Click me again</v-btn>

去除drop shadow  


1. <v-btn flat dark color="pink">Click me again</v-btn>  
   使得背景透明，悬停的时候有粉红效果，因为现在color不再控制背景色，而是控制字的颜色和悬停效果  
   
2. 带有icon的button, 这里的v-icon全部来自material design



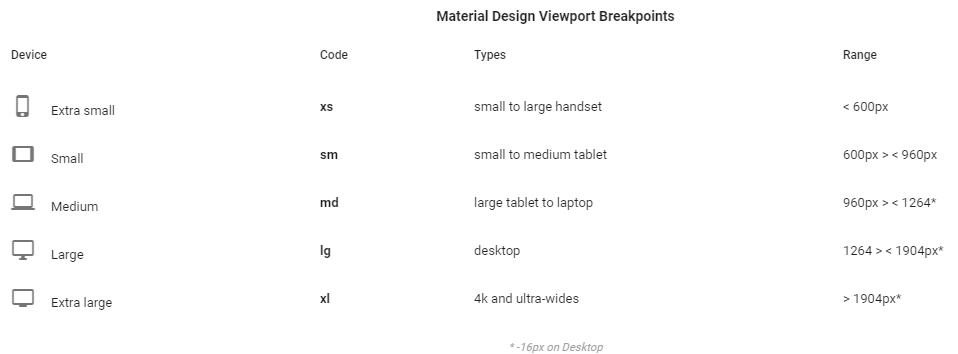


1. 圆形的button  
   



06\_breakpoints\_visibility

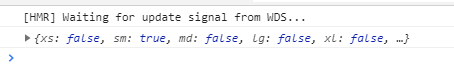
Vuetify has a 12 point grid system. Built using **flex-box**, the grid is used to layout an application's content. It contains 5 types of media breakpoints that are used for targeting specific screen sizes or orientations. The props for grid components are actually classes that are derived from their defined properties. This allows you to easily specify these helper classes as props, while still providing the classes to be used anywhere.



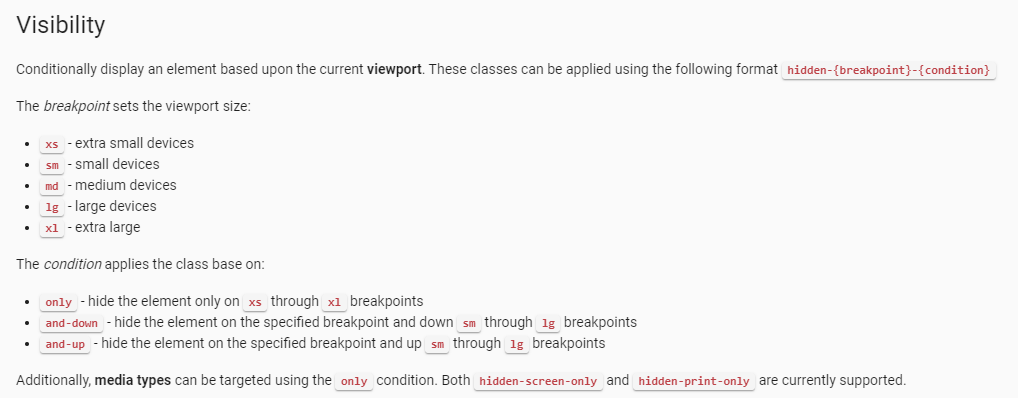
## **[Breakpoint object](https://vuetifyjs.com/en/framework/breakpoints" \l "breakpoint-object)**

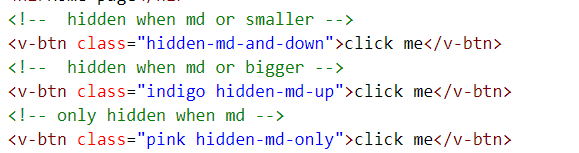
Vuetify converts the available breakpoints into an accessible object from within your application. This will allow you to assign/apply specific properties and attributes based upon viewport size. The object is available at **$vuetify.breakpoint**.





同样的visibility

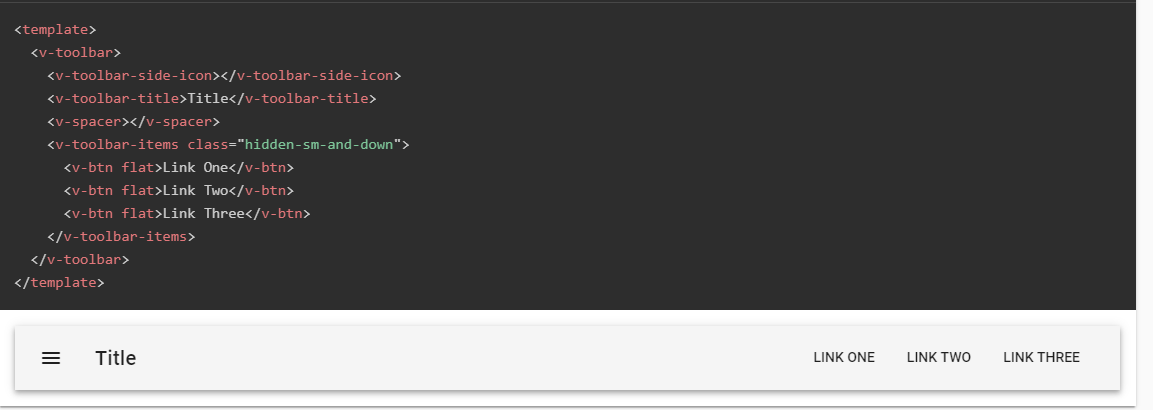


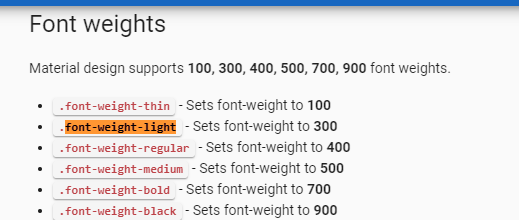


决定什么时候隐藏 hidden-<size>-<condition>

07\_toolbars

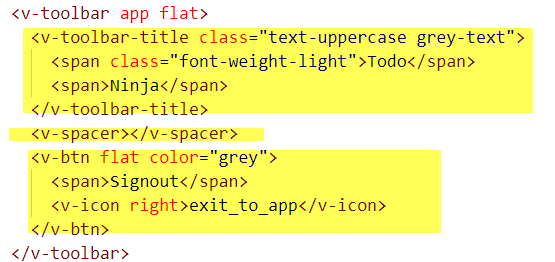
A toolbar is a flexible container that can be utilized in a number of ways. By default, the toolbar is 64px high on desktop, 56px high on mobile portrait and 48px high on mobile landscape. There are a number of helper components available to use with the toolbar. These are **v-toolbar-side-icon**, **v-toolbar-title** and **v-toolbar-items**



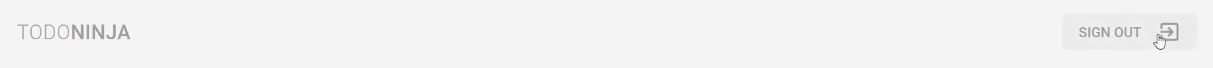


v-spacer

很有用的component, take all the available room and space out 2 sibling items

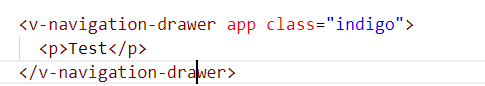


就是把我们的v-toolbar-title 和 v-btn分别推到左右端



08\_navigation\_drawers

最简单的一个drawer

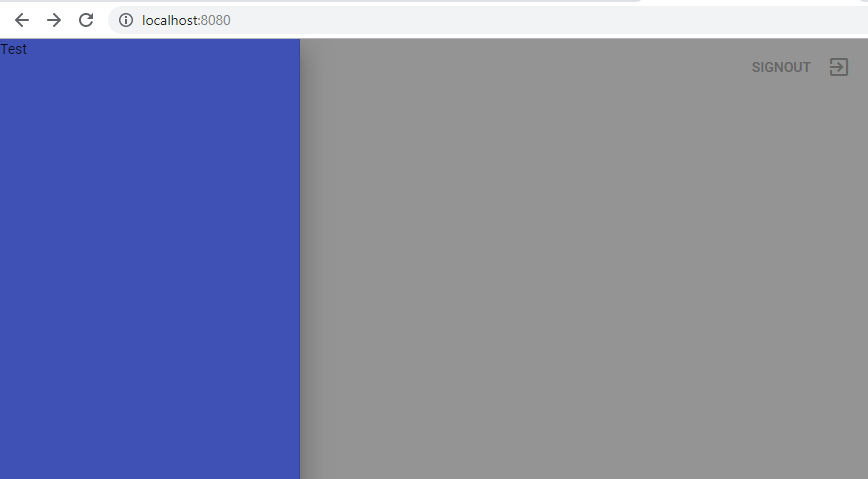


但是默认它不会显示

加上最简单的控制，默认就直接显示



此时屏幕直接打开drawer,只有点击右边才会消失



显然这不够好，我们需要一个按钮，点击才会显示drawer



09\_themes

如何创建或者定义自己的color theme?

打开/plugins/vuetify.js, 里面是vuetify默认的setup

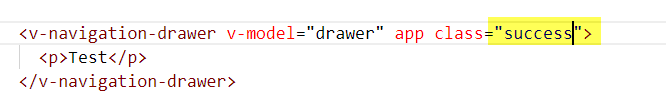
我们可以提供我们的theme

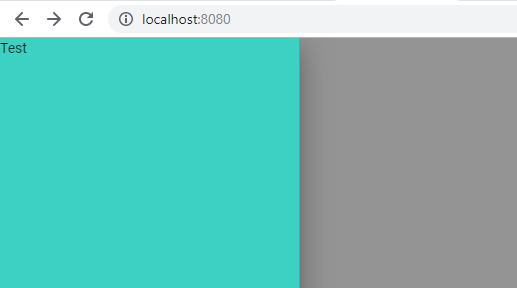


Change to:



此时默认的theme就被覆盖了

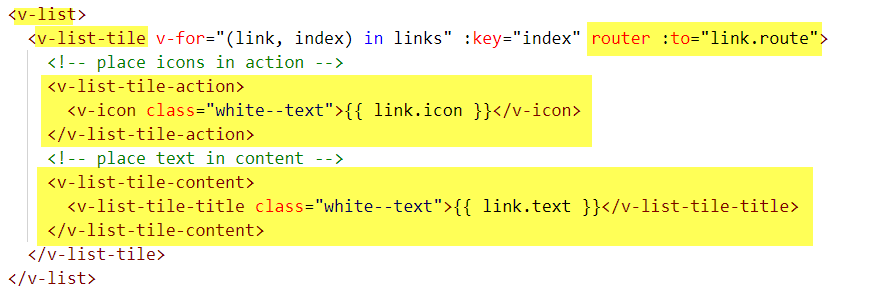




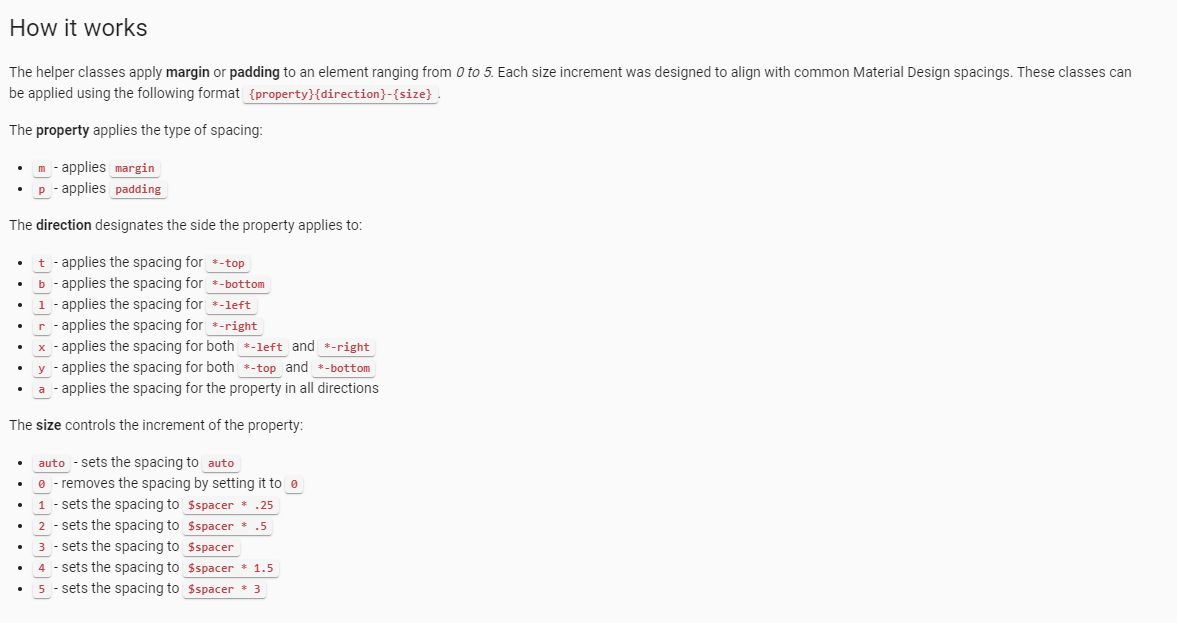
10\_lists

结构相对比较复杂：

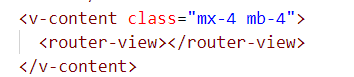
1. 最外面套v-list
2. 每一项使用v-list-tile
3. 每一项的icon放在v-list-tile-action里面
4. 每一项的content放在v-list-tile-content里面
5. 需要使用route的element，可以直接加上’route’ 和 ‘:to’ property 即可，非常非常方便



12\_padding\_margin



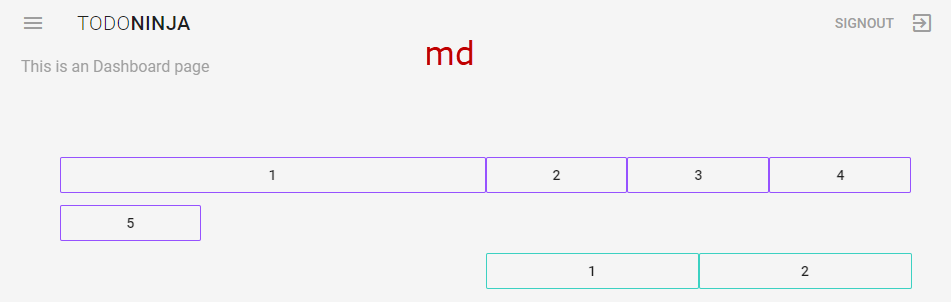
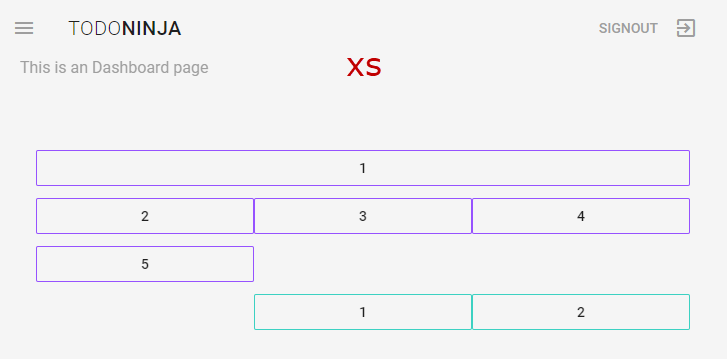
例子:

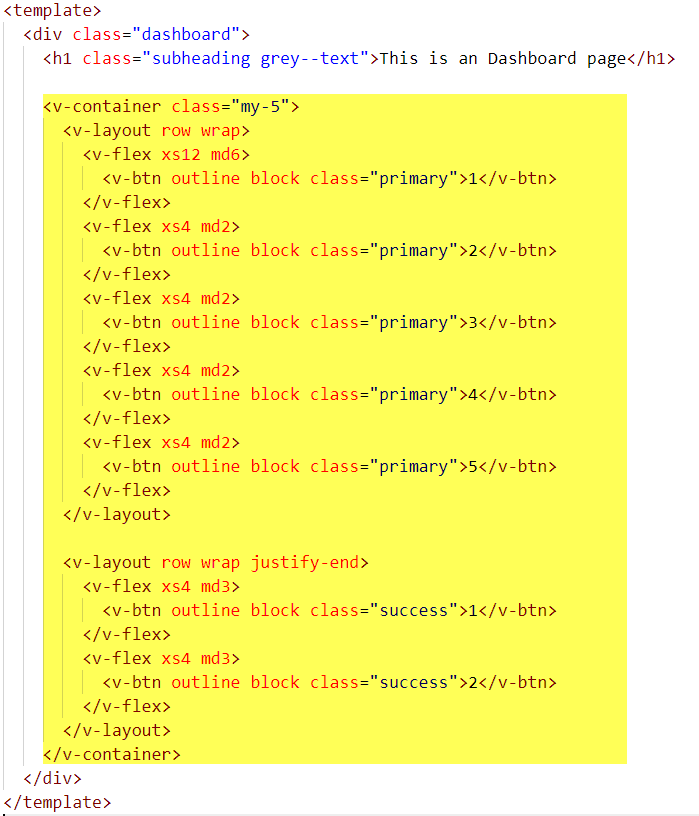


左右margin 4, margin bottom 4

13\_the\_grid\_system\_part\_1

1. The **v-container** can be used for a center focused page, or given the **fluid** prop to extend its full width.
2. **v-layout** is used for separating sections and contains the **v-flex**. The structure of your layout will be as follows, **v-container** » **v-layout** » **v-flex**. Each part of the grid chain is a flex-box element.
3. The final, **v-flex**, automatically sets its children to have **flex: 1 1 auto**.
4. 每个v-layout都是一个flex row 或者column, 里面用于放flex element

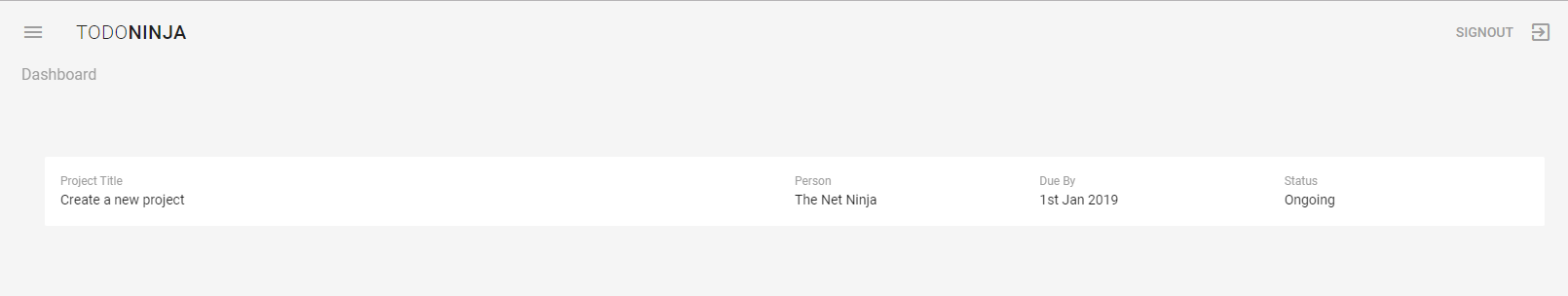
 



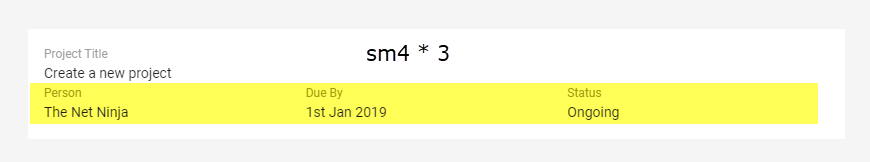
14\_grid\_system\_part\_2



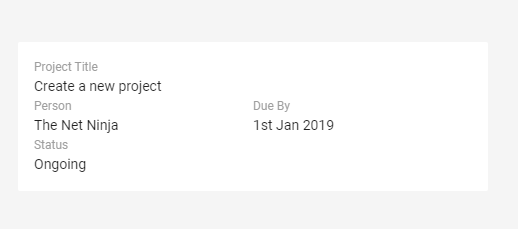
在md下面处以同一行



Sm下面分成两行

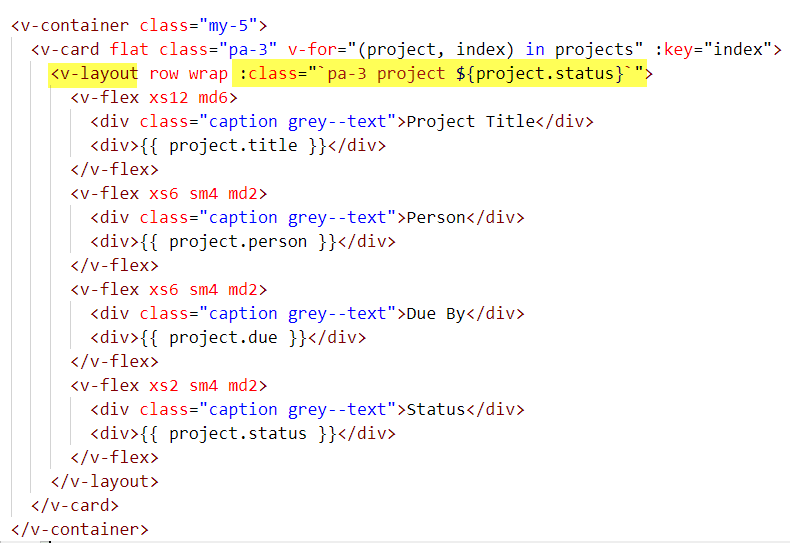


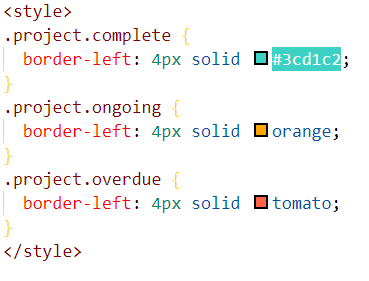
Xs下面分成3行

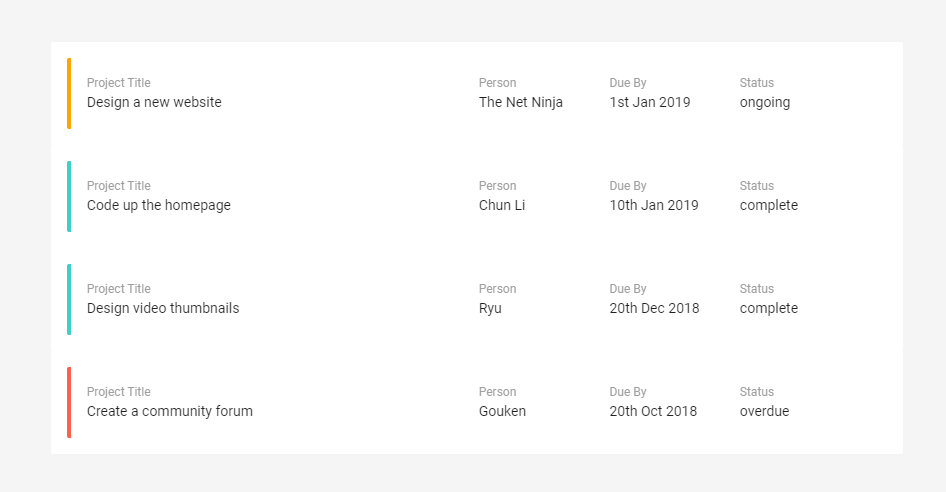


15\_dummy\_project\_data

本章最大的收获就是，v-layout实际上会被render成为 div.layout, 所以可以对它使用各种css

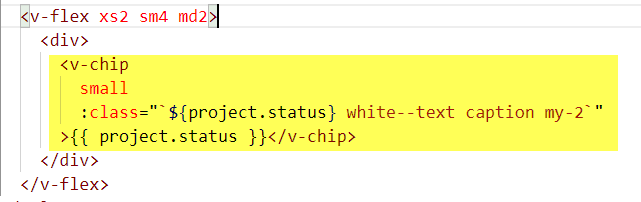






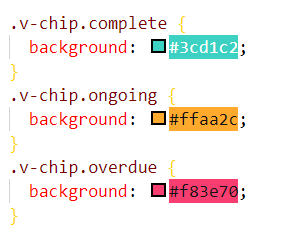
16\_chips

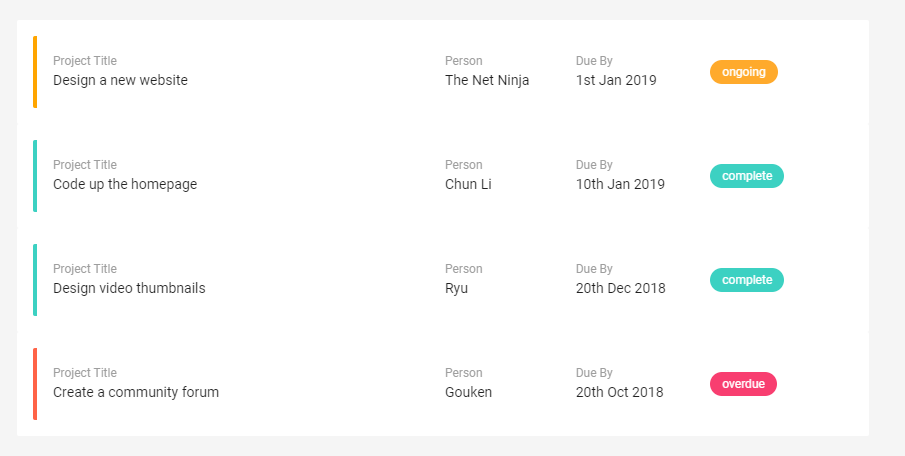
用v-chip 包裹即可



注意，实际上v-chip会被render 成为span.v-chip

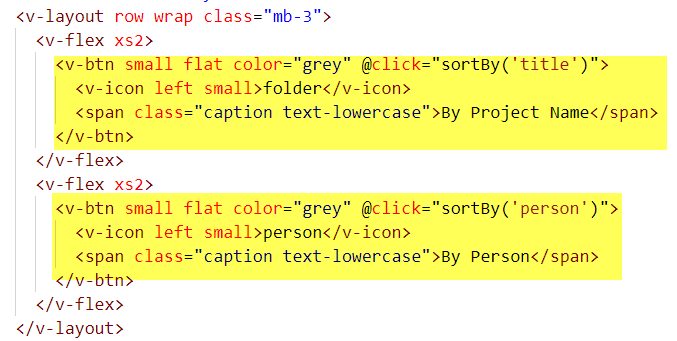
所以可以使用.v-chip来作为selector



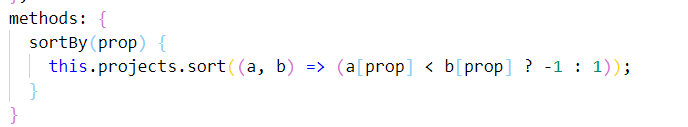


17\_sorting\_projects

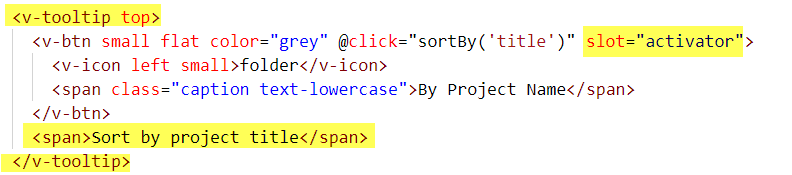
本章学到了一个不错的sort思路



然后是sortBy方法



18\_tooltips



这几个都不能少，其中slot=”activator” 是parent使用named slot, 传递给slot name=”activator” 的

When you declare a Vue component, you can create [**Named Slots**](https://vuejs.org/v2/guide/components.html#Named-Slots), which is a **Vue *native*** feature (not from Vuetify):

For example, suppose we have an app-layout component with the following template:

<div class="container">

<header>

<slot name="header"></slot>

</header>

<main>

<slot></slot>

</main>

<footer>

<slot name="footer"></slot>

</footer>

</div>

Parent markup:

<app-layout>

<h1 slot="header">Here might be a page title</h1>

<p>A paragraph for the main content.</p>

<p>And another one.</p>

<p slot="footer">Here's some contact info</p>

</app-layout>

The rendered result will be:

<div class="container">

<header>

<h1>Here might be a page title</h1>

</header>

<main>

<p>A paragraph for the main content.</p>

<p>And another one.</p>

</main>

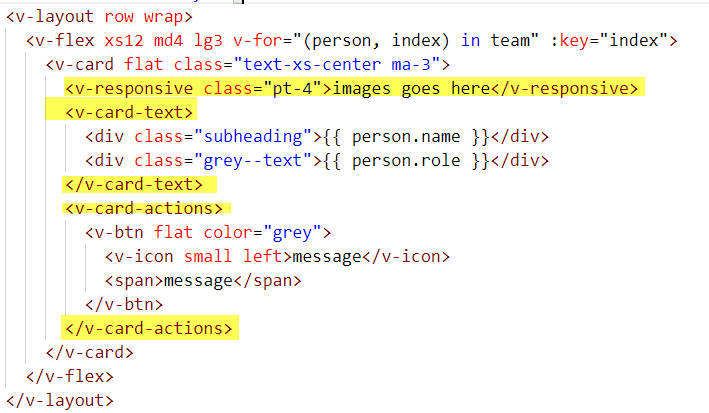
<footer>

<p>Here's some contact info</p>

</footer>

</div>

19\_cards

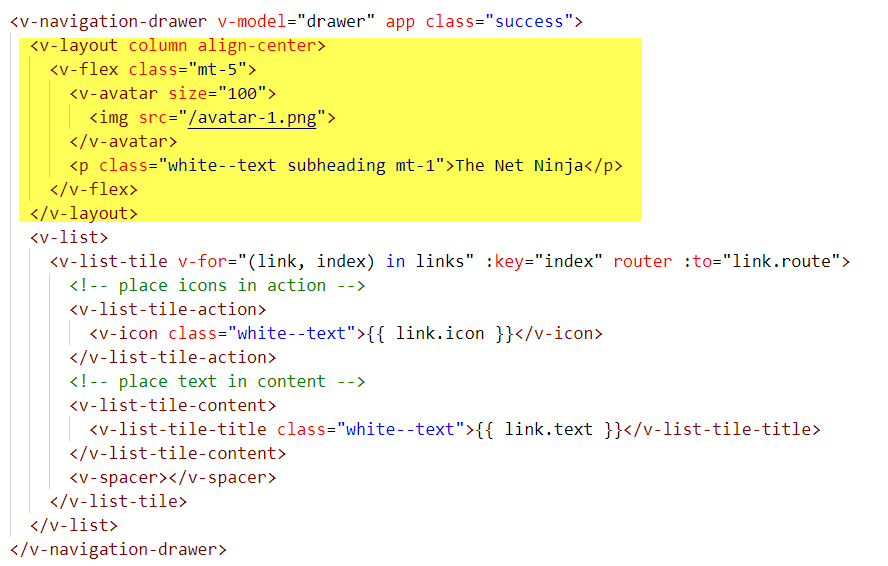


20\_avatars

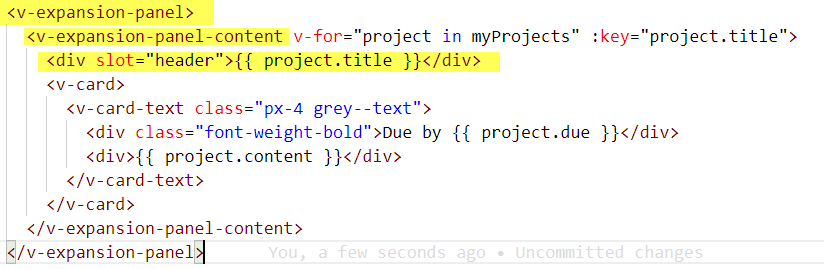
Team.vue



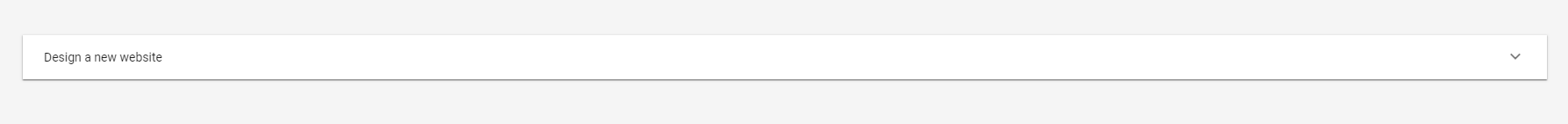
以及用在drawer



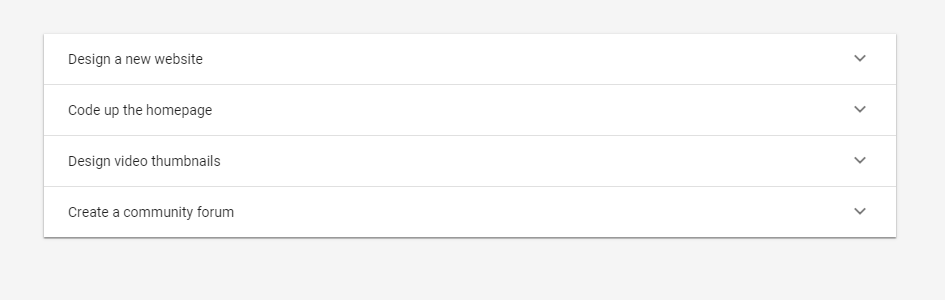
21\_expansion\_panels



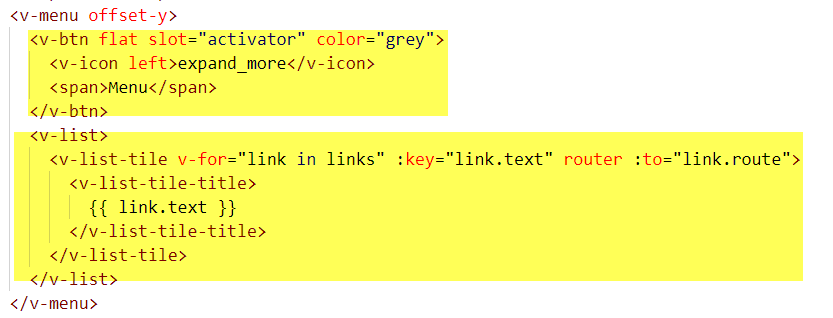
其中<div slot="header">{{ project.title }}</div> 显示的是未展开的时候的标题



另外注意循环是循环在v-expansion-panel-content



22\_menus.mp4



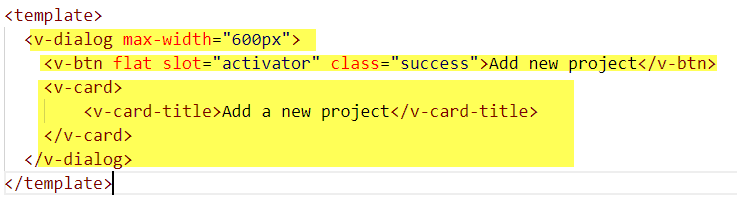
基本上menu分为两个部分，1个是没有展开的状态，是一个button, 另外一个是一个list, 里面放你要放的items

注意v-menu 的offset-y属性是把展开的list items往下移动一点，不要盖住button

23\_popups\_dialogs

出乎意料地简单

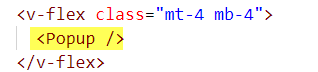
Dialog本身分为两个部分



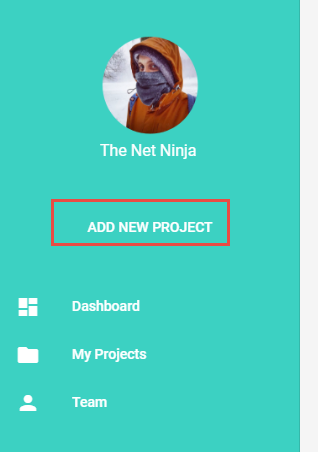
1. 一个btn, 显示没有显示出dialog时候的状态
2. 一个 v-card, 显示激活的状态

但是这是个component, 需要被点击激活

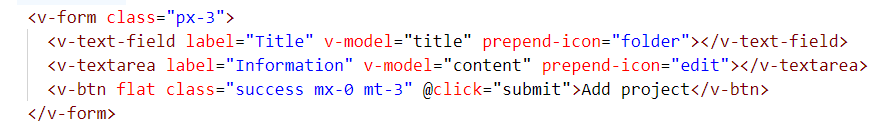
我们把它放在一个别的component,



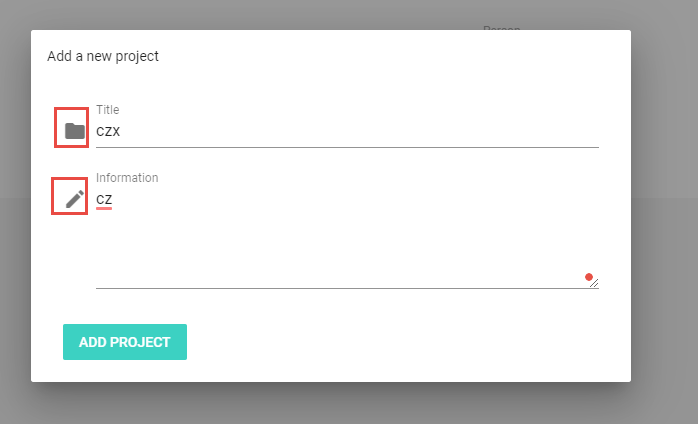
此时点击这个按钮，即Popup未激活的状态时候的那个按钮，就可以显示dialog



24\_form\_basics

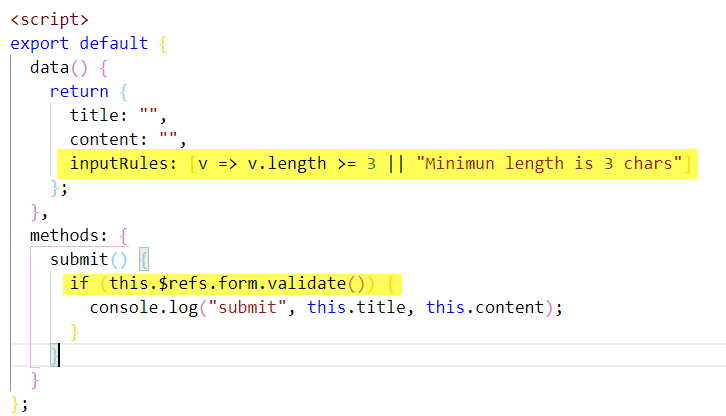


这个prepend-icon就是显示在field前面的icon

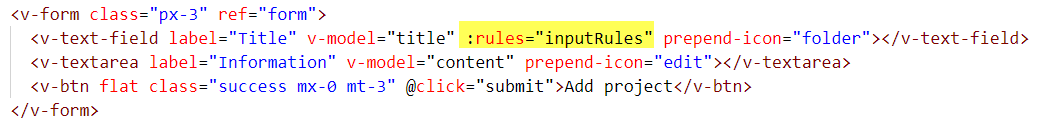


26\_simple\_form\_validation

首先要制定rules for validation, 一定要是个array



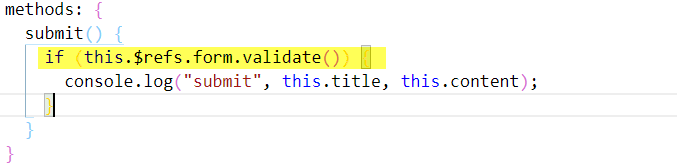
然后在input 上面使用

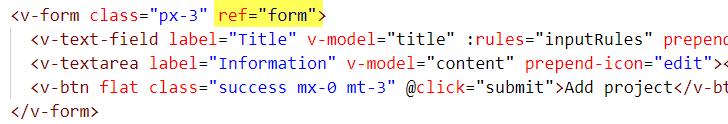


此时就有validation了



此时还有一个文件，就是submit时候的验证





这个就是找一个叫做form的ref,找到以后运行validate()，检查这个form里面的fields是否符合rules