# Vue--常用指令总结

## 什么是vue？

Vue.js是JavaScript MVVM（Model-View-ViewModel）库，十分简洁，Vue核心只关注视图层，相对AngularJS提供更加简洁、易于理解的API。Vue尽可能通过简单的API实现响应的数据绑定和组合的视图组件。是现阶段前端中比较火的框架之一，我们的大麦项目就采用这一框架来实现,在使用vue实现项目的过程中，不得不说的就是vue 当中的指令，接下来我们就结合我们的项目对vue中的指令进行总结，这里并不会对vue中全部的指令进行介绍，我们只是对大麦项目中用到的指令进行介绍，如果大家还想学习更多的关于vue指令方面的知识，可以取vue官方网站进行学习（https://cn.vuejs.org/v2/guide/conditional.html），另外我们这里声明一下，我们项目中使用的版本是2.0的版本，之后的所有介绍都是基于2.0的版本进行的，下面我们就正式进入vue指令的介绍；

## 什么是vue指令？

**vue的指令是以v-开头的，它们作用于HTML元素，指令提供了一些特殊的特性，将指令绑定在元素上时，指令会为绑定的目标元素添加一些特殊的行为，我们可以将指令看作特殊的HTML特性（attribute）。**

## 为什么使用vue指令？

**最重要的原因就是使用vue指令可以帮我们简化操作，可以更加方便的完成一些业务代码，例如我们之前传统开发当中的条件判断，一定要写到JavaScript中才能完成，但是现在在vue中可以使用v-if指令就可以完成，那我们项目中都用到了哪些vue指令呢，下面用一张图给大家展示**



**下面我们就来介绍下项目中用到的这些vue的内置指令：**

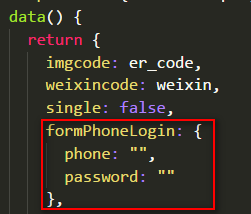
## v-model指令

v-model 指令，它能轻松实现表单输入和应用状态之间的双向绑定。v-model这个指令只能用在<input>, <select>,<textarea>等这些表单元素上，所谓双向绑定，指的就是我们在js中的vue实例中的data与其渲染的dom元素上的内容保持一致，两者无论谁被改变，另一方也会相应的更新为相同的数据。

经过上面对于v-model指令的介绍，大家可以知道，v-model主要是应用到表单域中使用，可以比较轻松实现双向数据绑定，能够较为轻松的拿到表单当中的值，提到表单的我们最先想到的是表单应用较为广泛的登录页面和注册页面，下面我们以登录页面为例：







在data中，我们声明formPhoneLogin对象对手机号码和登录密码进行接收；

**注：**

**1、这里的v-model需要在表单的地方才能使用**

**2、在表单中一般都是需要获得用户信息的，在这个时候通常会用到v-model进行双向数据绑定**

## v-if指令

V-if 也被称为条件渲染指令，它根据表达式的真假来插入和删除元素，具体使用情况可以查看下面的语法：

v-if="expression";

expression是一个返回boolean值的表达式，这里的表达式可以是一个boolean属性（true或者false），也可以是一个返回boolean的运算式。如果expression的值为true才会被渲染出来；

条件渲染指令用的还是比较多的，比如项目中有遇到条件判断时候就可以使用v-if来实现一些简单的条件判断，这里我们同样选择我们项目中的使用v-if指令比较典型的地方进行讲解；

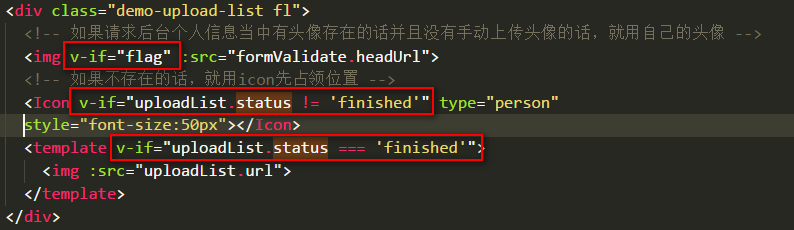
这里我们以上传头像为例进行说明v-if指令的使用：



这里需要给大家梳理一下业务流程：

1. 在初次进入到这个头像页面的时候，肯定要先确定用户是否上传过头像，如果上传过用户头像，肯定这里显示的就是用户上传的头像
   1. 如果是这种情况的话，用户再次上传头像，目的明确是修改用户头像，这时候显示的应该是用户重新上传的头像，也就是修改之后的头像
2. 如果用户之前并没有进行头像的上传，那么这里肯定显示的就是一个占位符，并没有用户头像进行显示
   1. 这时候，如果用户上传头像就肯定显示用户的头像了

通过上面业务流程的梳理，这里的条件判断是比较多的，不同情况下显示不同的图片，这里我们就可以用v-if指令完成了



这里有3个v-if，接下来我们依次来分析他们的作用：

第一个v-if：这里既是一开始进入页面，先去后台数据库获取用户的信息，如果有头像的图片，那么这里的flag的值为true，这里的头像也就显示用户已经上传的头像，相反，如果没有头像信息，这里flag为false，这里第一个v-if对应的img标签也就不会显示；

第二个v-if：这里对应的情况就是用户没有上传过头像，是第一次来到这个头像页面，在用户没有上传头像的情况之前，uploadList.status!=finished 都是true，这是显示的也就是头像的一个占位符，如果这个时候用户上传了头像 表达式 uploadList.status 就是finished 状态了，uploadList.status!=finished 就是false了，占位符的图片也就不会显示了，也就是第二个v-if对应的img标签也就不会显示了

第三个v-if：这里不管是用户之前上传了头像还是之前一直没上传过头像，只要是用户再次点击上传头像的话，uploadList.status===finished的值都是true，也就是这里显示的都是用户再次上传的头像，用户不上传的话uploadList.status===finished的值都是false，同理这个时候，第三个v-if对应的img不会显示出来

这个头像上传的案例可以说是比较复杂一点的条件判断语句，大家如果可以理清楚这里的业务并可以实现这里的条件判断，肯定v-if这里指令就不会有问题了；

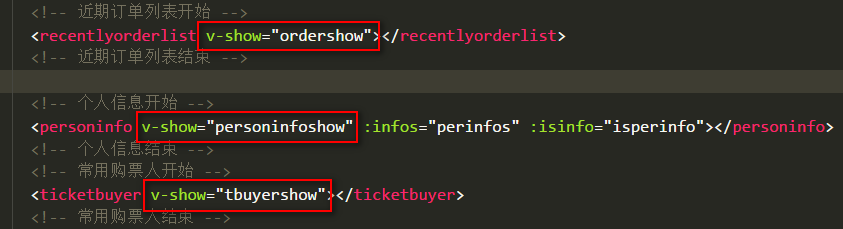
## v-show指令

v-show也是条件渲染指令，和v-if指令不同的是，使用v-show指令的元素始终会被渲染到HTML，它只是简单地为元素设置CSS的style属性。不满足条件的元素被设置了style="display:none"样式，如果是v-if不满足条件的情况下，是不会渲染到html中的；

下面我们以项目中的实际运用为基础进行说明v-show指令的使用，这里首先要给大家理清楚业务逻辑，然后才能更好的去讲解v-show指令的使用：



这里的页面是我们的个人信息页面，这里面包含三个页面，可以看到左边使用方框圈起来的选项，当我们点击的时候，需要切换到相应的页面，比如这里我们点击常用购票人的时候，需要跳转到常用购票人的页面，这里我们就可以使用V-show指令进行切换显示不同页面



这里我们引入三个组件，也就是代表三个页面，分别是订单页、个人信息页、常用购票人页面，当我们点击订单管理的时候，我们会把ordershow变量设置为true，而其余的两个变量我们会设置为false，此时订单的页面也就会显示出来，而其余的两个组件会被添加style="display:none"，也就是隐藏起来了，这样也就实现了点击不同文字按钮，可以切换不同页面了；以上部分便是v-show指令的用法。

到这里的话，我们有必要给大家讲解一下v-if和v-show的区别，同样是条件渲染指令，那他们的区别有哪些：

**注意：v-if/v-show区别**

一般来说，v‐if 有更高的切换消耗而 v‐show 有更高的初始渲染消耗。因此，如果需要频繁切换 v‐show 较好，如果在运行时条件不大可能改变 v‐if 较好

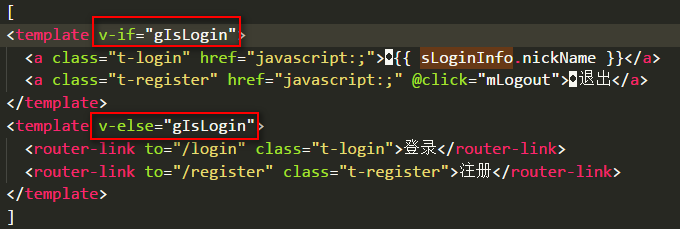
v-if当条件成立的时候会将元素加上，不成立的时候，就会移除dom，并且内部的指令不会执行；

v-show只是简单的隐藏和显示；

## v-else指令

可以用v-else指令为v-if添加一个“else块”。v-else元素必须立即跟在v-if元素的后面——否则它不能被识别。 v-if为true，后面的v-else不会渲染到HTML。

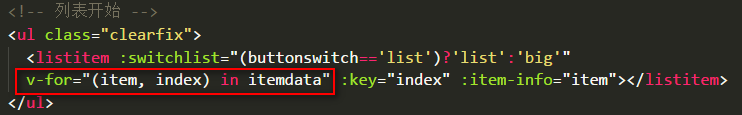
下面我们就以是否登录来给大家做讲解,先给大家看一下代码：



这里需要判断gIsLogin是否为true，如果是true的话，说明用户已经登录成功了，这种状态下，v-else对应的template就不会显示了，这里因为是使用的v-if，所以是不会渲染到html中的，当gIsLogin为false时候，也就是意味着用户没有登录，那么这时候v-else对应的template的内容就会对应的渲染到html中，也就是页面上会显示用户登录和注册的按钮；这时v-if对应的template的内容也就不会渲染到html中，这里需要提醒大家的是这里的v-else指令就是非此即彼的判断，如果是过于复杂的判断，就不能使用v-else指令了。

## v-for指令

**v-for指令基于一个数组渲染一个列表，它和JavaScript的遍历语法相似,v-for="item in items" items可以代表数组，item是当前被遍历的数组元素项。这个指令用的频率是相当高的，比如我们的商品列表页面，一条一条的商品列表肯定是使用循环遍历出来的，还有就是常用购票人列表，肯定也是拿到后台数据库的数据，进行循环遍历显示的，这里循环遍历显示的操作，在实际项目中用的还是比较多，大家一定要重点掌握一下，下面我们就以商品列表循环为例，给大家说明一下v-show指令的具体使用：**



这里给大家解释下这个这段代码，首先（item，index），其中item代表遍历的项目，index代表当前被遍历项目的索引值，这里的索引值是唯一的，itemdata是我们的数组对象，也就是我们遍历的数据集合，这样的话，就会使用itemdata里面的数量来遍历渲染每一个列表到页面中显示

**注：**

如果遍历的对象中没有唯一的键供追踪 那么在2.0版本的话应该用 :key="index"，这里说下，加上：key有利于列表渲染，在实际操作的时候，尽量添加:key="index"

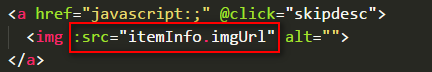
## v-bind指令

v-bind指令可以在其名称后面带一个参数，中间放一个冒号隔开，这个参数通常是HTML元素的特性（attribute），v-bind是动态绑定指令，默认情况下标签自带属性的值是固定的，在为了能够动态的给这些属性添加值

可以使用v-bind:你要动态变化的值="表达式"

例如：v-bind:class

下面我们就动态的显示图片来作为案例进行讲解，比如我们的商品列表中商品的图片，这是需要每个商品的图片是不一样的，这是我们就可以使用v-bind指令进行动态的绑定：



图中标红的为使用v-bind进行绑定img标签的src属性，这样的话就可以动态的给这些属性添加值了，当不同商品进行渲染的时候，也就可以渲染出不同的图片了

**注：**

这里需要明确一下v-bind指令的使用场景，默认情况下标签自带属性的值是固定的，在为了能够动态的给这些属性添加值，可以使用v-bind，比如上面的src属性，还有比如class属性等等

另外大家可能已经注意到了v-bind指令可以使用 :号 的方式进行简写,不要使用{{src}}进行设置链接，这样会报错的；

## v-on指令

v-on指令用于给DOM元素添加监听事件，它的用语法和v-bind是类似的，在具体的项目当中用的是非常多的，例如监听元素的点击事件，在我们书写项目当中的业务逻辑代码的时候，都会牵扯到事件的操作，这个时候我们就会使用到v-on指令了，下面我们就拿项目中的点击事件进行给大家介绍，来讲解v-on指令

例如我们的用户填写完订单之后，需要点击提交订单的按钮来确认自己的订单，这就是我们最为常见的一种监听dom点击事件的操作了；

看下面代码：

<Button type="primary" :disabled='disabled' @click="submitorder">同意以上协议并提交订单</Button>

标红的部分可以写成v-on:click="submitorder",代码部分是v-on指令的一个简写形式，当我们点击按钮的时候，会触发submitorder这个方法去执行，至于这个方法的执行部分，我们需要把它写到methods方法当中去，这样的话，点击按钮就会去触发这个提交订单的方法执行；

**注：**

1、v-on指令，就是当我们需要有监听任何事件的时候，都要用到的指令

2、v-on 我们可以使用 @号 的方式进行简写

## 总结

以上vue指令是我们项目中用到的，也是vue指令中经常用到的指令，如果大家有兴趣了解更多的指令方面的知识点，可以自行去vue的官方网站去学习。

vue官方网站地址：<https://cn.vuejs.org/v2/guide/conditional.html>