Vue2.0 在官方网站<https://cn.vuejs.org>下个就行

Vue版本差异还是有的 我现在用的是2.1.10感觉还可以

Vuejs已经下好了之后 为了能启动项目 需要安装一个liver-server 起服务用的 前提是已经安装了nodejs这个很好安装<http://nodejs.cn/download/>

命令行安装live-server

Npm install –g live-server

如果npm install -g live-server安装报错就用npm install live-server -gf安装

在项目目录下 live-server

然后就可以开始写项目了

一 基本指令



v-if 和v-show的区别：

v-if： 判断是否加载，可以减轻服务器的压力，在需要时加载。

v-show：调整css dispaly属性，可以使客户端操作更加流畅。

v-for:

循环指令，循环就不用说了，item in items 但不像ng能直接通过orderby直接对循环进行操作 vue需要手动写

在computed里面写 这个是在数据之前进行计算 可以说对数据的操作 让他排序啊 改变啊 什么的都可以 注意 在vue里面调用对象的时候用this直接调用就行 有的时候会忘写 所以我提醒自己一下

在vue里写函数的时候 直接在外面写就行 在vue作用域中可直接调用

实例如下：­



v-text and v-html

我们已经会在html中输出data中的值了，我们已经用的是{{xxx}},这种情况是有弊端的，就是当我们网速很慢或者javascript出错时，会暴露我们的{{xxx}}

如果在javascript中写有html标签，用v-text是输出不出来的，这时候我们就需要用v-html标签了。

双大括号会将数据解释为纯文本，而非HTML。为了输出真正的HTML，你就需要使用v-html 指令。

需要注意的是：在生产环境中动态渲染HTML是非常危险的，因为容易导致XSS攻击。所以只能在可信的内容上使用v-html，永远不要在用户提交和可操作的网页上使用。



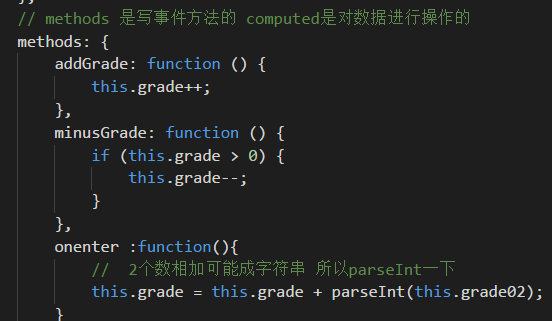
v-on事件监听器

就是绑定事件用的 可以是v-on:click 也可以是@click 都行

绑定键盘事件的时候可以直接绑定键值 @keyup.13=”onenter”

直接绑定方法名 不用（）

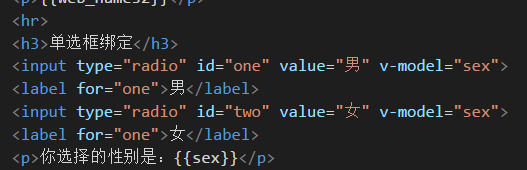






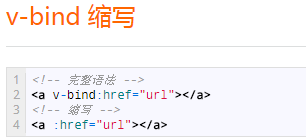
v-model 数据绑定



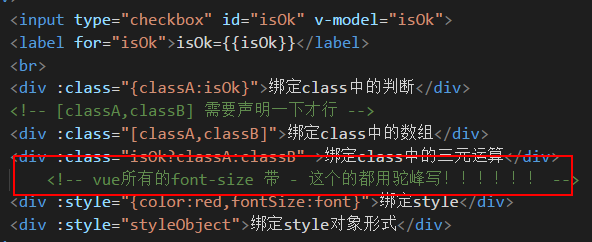


选框输入 都是绑定一个model然后初始化一个值 当鼠标点击的时候 点那个哪个就把值传给model达到数据绑定的效果

v-bind









\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

二 全局api

什么是全局API？

全局API并不在构造器里，而是先声明全局变量或者直接在Vue上定义一些新功能，Vue内置了一些全局API，比如我们今天要学习的指令Vue.directive。说的简单些就是，在构造器外部用Vue提供给我们的API函数来定义新的功能。

自定义指令 Vue.directive

自定义指令有五个生命周期（也叫钩子函数），分别是 bind,inserted,update,componentUpdated,unbind

bind:只调用一次，指令第一次绑定到元素时调用，用这个钩子函数可以定义一个绑定时执行一次的初始化动作。

inserted:被绑定元素插入父节点时调用（父节点存在即可调用，不必存在于document中）。

update:被绑定于元素所在的模板更新时调用，而无论绑定值是否变化。通过比较更新前后的绑定值，可以忽略不必要的模板更新。

componentUpdated:被绑定元素所在模板完成一次更新周期时调用。

unbind:只调用一次，指令与元素解绑时调用。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | bind:function(){//被绑定       console.log('1 - bind');  },  inserted:function(){//绑定到节点        console.log('2 - inserted');  },  update:function(){//组件更新        console.log('3 - update');  },  componentUpdated:function(){//组件更新完成        console.log('4 - componentUpdated');  },  unbind:function(){//解绑        console.log('1 - bind');  } |

**自定义指令中传递的三个参数**

**el:** 指令所绑定的元素，可以用来直接操作DOM。

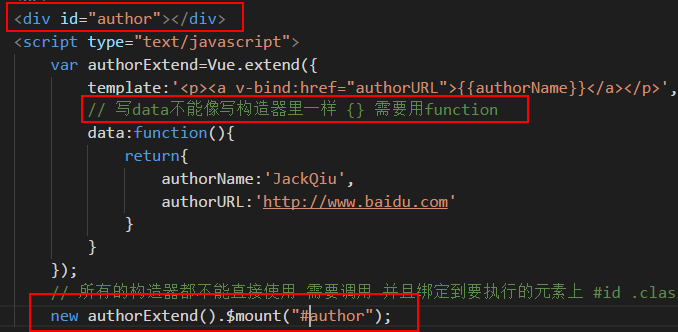
**binding:**一个对象，包含指令的很多信息。

**vnode:** Vue编译生成的虚拟节点。

Vue.extend 扩展构造器

**一、什么是Vue.extend？**

Vue.extend 返回的是一个“扩展实例构造器”,也就是预设了部分选项的Vue实例构造器。经常服务于Vue.component用来生成组件，可以简单理解为当在模板中遇到该组件名称作为标签的自定义元素时，会自动调用“扩展实例构造器”来生产组件实例，并挂载到自定义元素上。



Vue.set全局操作

Vue.set 的作用就是在构造器外部操作构造器内部的数据、属性或者方法。比如在vue构造器内部定义了一个count为1的数据，我们在构造器外部定义了一个方法，要每次点击按钮给值加1.就需要用到Vue.set。





App.$set(app.arr,1,”ddd”) ===🡺Vue.set(app.arr,1,”ddd”)

Template模板



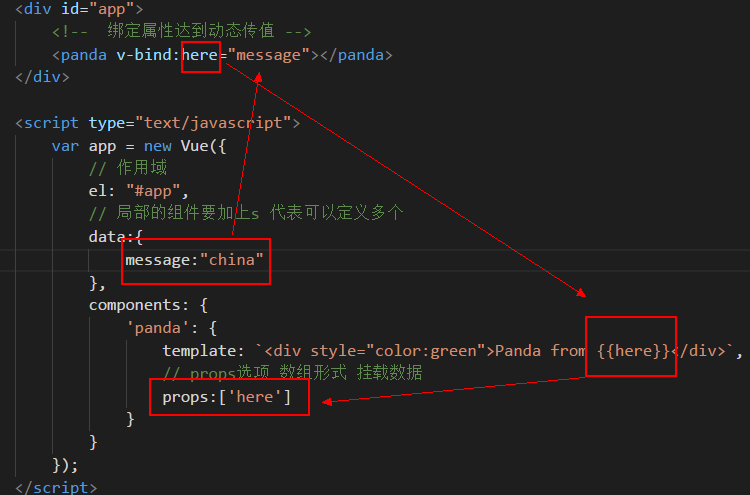
Component 组件 重要重要重要

component组件是Vue学习的重点、重点、重点，重要的事情说三遍。所以你必须学好Vue component。其实组件就是制作自定义的标签，这些标签在HTML中是没有的。比如：<jspang></jspang>，那我们就开始学习这种技巧吧。

指令构造的一般是属性 组件构造的一般是标签 也可以在标签里加属性 慢慢看



绑定属性



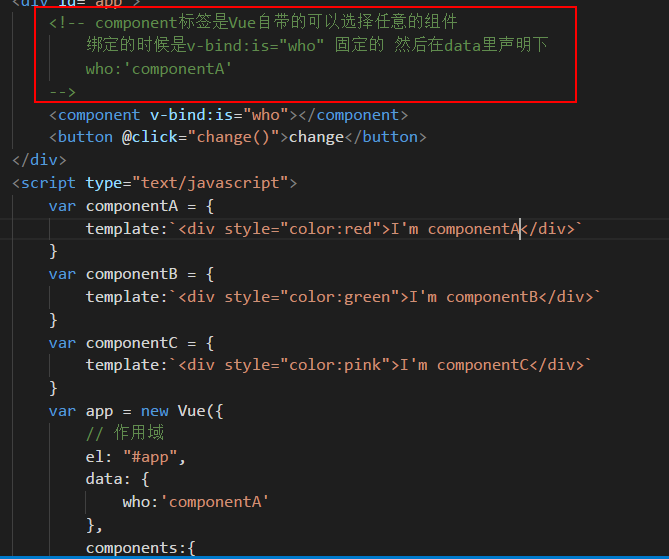
Component父子组件





子组件要在父组件上面声明才有效果

Component标签



三 vue2.0 选项

propsData

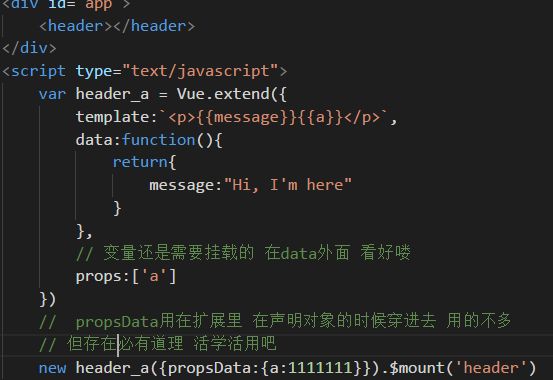
propsData 不是和属性有关，他用在全局扩展时进行传递数据

我们用propsData三步解决传值：

1、在全局扩展里加入props进行接收。propsData:{a:1}

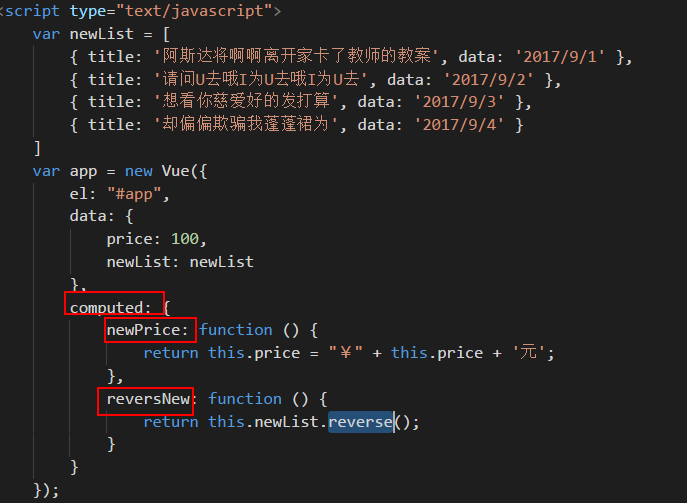
2、传递时用propsData进行传递。props:[‘a’]

3、用插值的形式写入模板。{{ a }}



Computed 计算

computed 属性是非常有用，在输出数据前可以轻松的改变数据

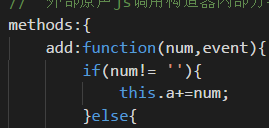


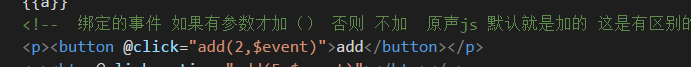
Method 方法选项

1. methods中参数的传递

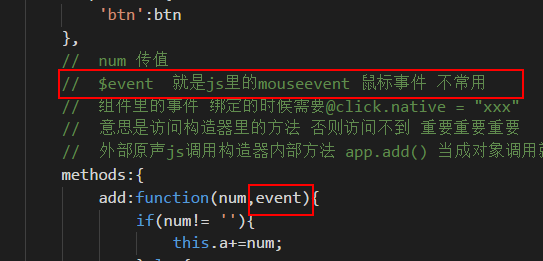
1、在methods的方法中进行声明，比如我们给add方法加上一个num参数，就要写出add:function(num){}.

2、调用方法时直接传递，比如我们要传递2这个参数，我们在button上就直接可以写。<button @click=”add(2)”></button>.

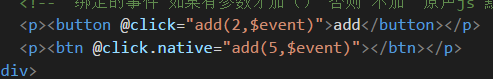


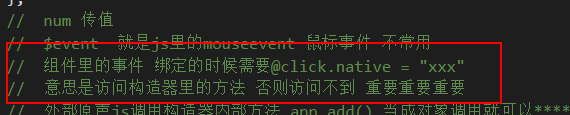


1. methods中的$event参数



1. native 给组件绑定构造器里的原生事件。





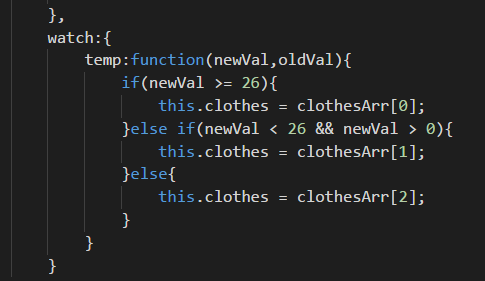
1. 作用域外部调用构造器里的方法





Watch 监控数据



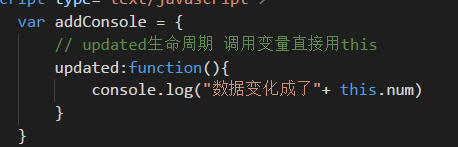


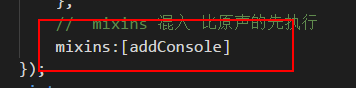
两种方法都可以的

Mixins 混入选项

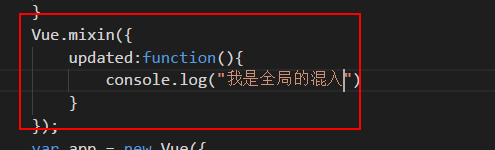
1、在你已经写好了构造器后，需要增加方法或者临时的活动时使用的方法，这时用混入会减少源代码的污染。

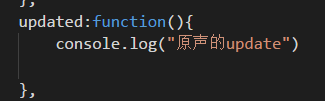
2、很多地方都会用到的公用方法，用混入的方法可以减少代码量，实现代码重用。

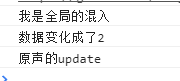




全局的



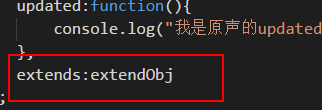
原声的



执行的先后顺序

Extend 扩展选项

跟mixins差不多 就是引入的时候是对象 不是数组



不管是混入还是扩展里 如果跟构造器的方法名是一样的是不会执行的

Delimiters



实例和内部组件

1 挂载jq 和在外部调用内部方法

