Vue的基本用法

软通大学
ISOFTSTONE UNIVERSITY

本节目标

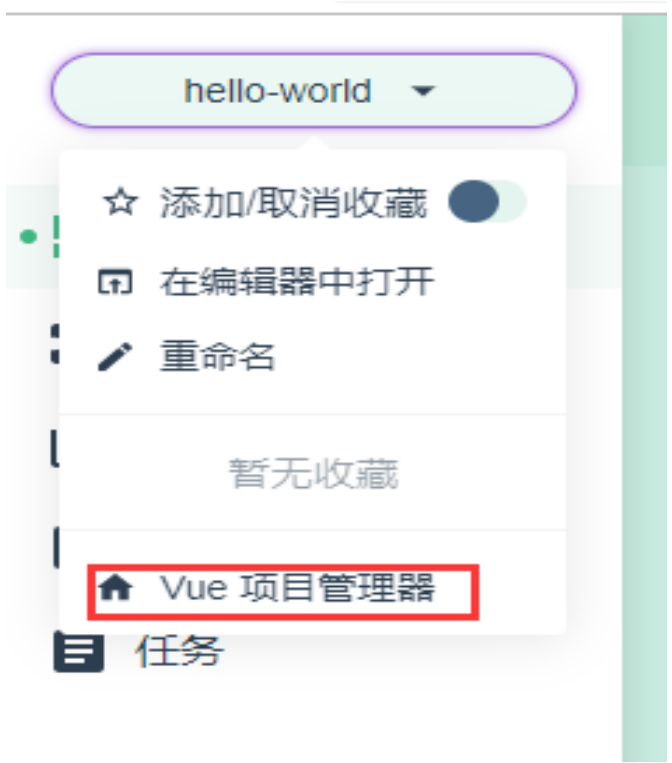
- 编写vue对象的基本代码结构
- 解决默认导入vue库的问题的三种方式
- 修改webpack项目配置的方法
- 插值表达式的使用
- 掌握全部vue指令的使用

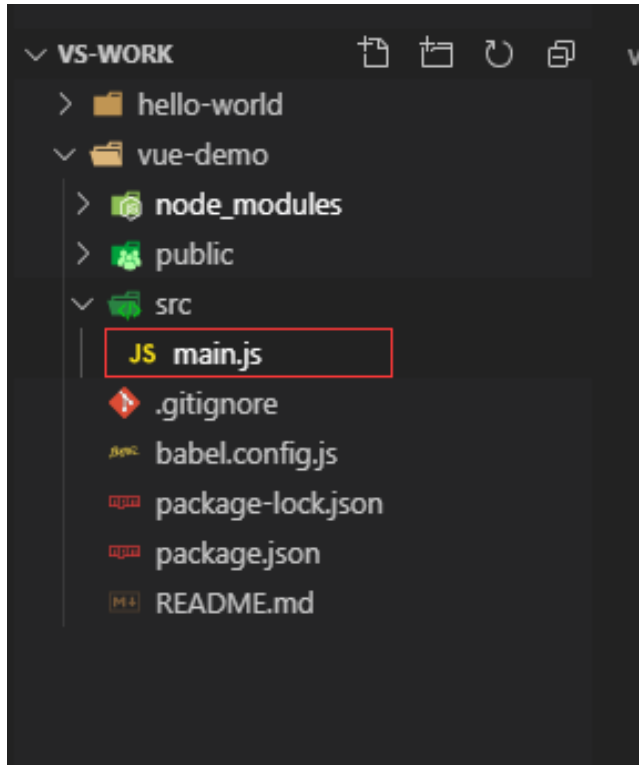
目录 CONTENTS

01 基本代码结构

基本代码结构

vue ui 创建一个新的工程，删除src目录无关内容：components目录、assets目录，app.vue文件(组件内容在后面章节作为专题讲解)，只保留main.js文件





基本代码结构

main.js 重新编写如下代码:

```
import Vue from '../node_modules/vue/dist/vue'

var vm = new Vue({
  el:"#app",
  data:{
        msg:"欢迎学习vue"
  }
})
```

- 1-vm对象就是调度者VM, data就是M, data数据更新驱动视图更新
- 2-el指定当前 Vue 实例，要控制页面上的哪个区域
- 3-data属性中，定义el中会用到的数据
- 4-msg其中一个数据，msg数据变化，el中的msg数据变化，不再手动操作dom

? 为什么要更改默认import Vue from “Vue”, 不修改会发现视图中的数据不能更新, 因为默认的import导入的实际上是 vue.runtime.common.js，功能不完整，只提供了 runtime-only 的方式。

解决方案有三种方式;

1. import Vue from '../node_modules/vue/dist/vue', 显示导入vue.js
2. 修改node_module/vue项目中package.json的main属性，入口文件修改为【“main”: “dist/vue.js”】, 不可取
3. 在项目的webpack.config.js中添加resolve属性, 但VueCli3.x项目内置了webpack, 修改它的默认配置相当困难而且不可取

基本代码结构

调整VueCli3.x项目内置webpack配置的方式:

VueCli3.x内置了webpack,修改内置webpack.config.js文件是不可取的, 查看官方文档:
https://cli.vuejs.org/zh/guide/webpack.html
可以得知,如果要调整webpack配置，在项目根目录下创建一个文件：vue.config.js，并对它进行调整配置，最终配置对象会被webpack-merge 合并入最终的 webpack 配置。实际上项目中的webpack配置多如牛毛，这也是VueCli3.x为我们内置webpack的初衷，但是在实际项目开发中，难免我们会需要调整项目的某些配置，这时候就通过vue.config.js文件进行调整配置，而vue.config.js的配置该怎么配?就比如现在要修改vue的默认导入路径，github提供了开源项目，最大程度上帮助我们调整webpack配置: **https://github.com/staven630/vue-cli3-config**


README.md

vue-cli3 全面配置(持续更新)

细致全面的vue-cli3配置信息。涵盖了使用vue-cli开发过程中大部分配置需求。

不建议直接拉取此项目作为模板，希望能按照此教程按需配置，或者复制vue.config.js增删配置,并自行安装所需依赖。

其他系列

★ Blog

★ Nuxt.js 全面配置

目录

- √ 配置多环境变量
- √ 配置基础 vue.config.js
- √ 配置 proxy 跨域
- √ 修复 HMR(热更新)失效
- √ 修复 Lazy loading routes Error: Cyclic dependency
- √ 添加别名alias
- √ 压缩图片

02 插值表达式

插值表达式

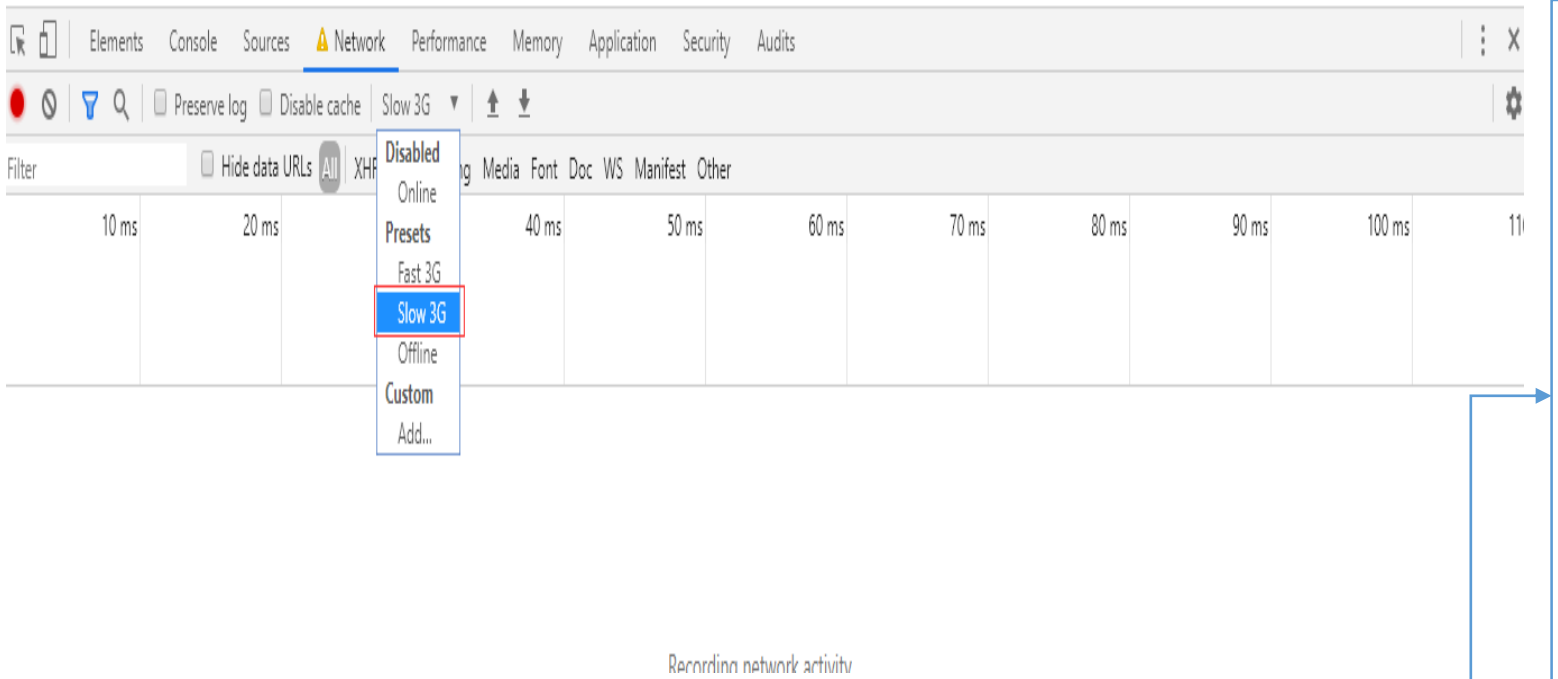
```
<!DOCTYPE html>
<html lang="en">
...
<body>
   <div id="app">
     <!-- 模板表达式: 又叫插值表达式，对应vm示例中的data对象的属性 -->
     {{msg}}
   </div>
 </body>
</html>
```

1：{{}} 官方称之为Mustache语法，只要定义在vm实例上的data对象中的msg属性发生了变化，插值处的内容会自动更新
2：{{}} 是支持Javascript表达式，合法的js表达式都可以再Mustach标签内运行，例如：
{{number + 1}}
{{ok ? ‘yes’ : ‘no’}} : ok为true，取yes, ok为false，取no
{{message.split(‘’).reverse().join(‘’)}} ：字符串运算
但是注意，它不支持语句和流程控制，所以{{}} 只建议做简单的js运算工作。

03 vue指令

v-cloak解决闪烁问题

当网速很慢的时候，vue将数据渲染到界面中，会出现插值表达式闪烁问题，比如，通过chrome浏览器模拟一下:打开F12开发者工具，作出如图调整:




```
<style>
   /* v-cloak配合属性选择器，默认隐藏，当数据回来的时候，v-cloak指令会自动转换显示模式 */
    [v-cloak]{
      display: none;
    }
</style>

<body>
  <div id="app">
        <p v-cloak>{{msg}} </p>
  </div>
</body>
```

再次启动项目访问就可以明显的感觉到闪烁问题了，使用v-clock指令可以解决这个问题

软通大学 ISOFTSTONE UNIVERSITY

v-text与v-html

v-text与v-html 和 jquery中的text()与html()方法是一样的作用，都可以设置标签的文本内容，text纯文本设置，html可以设置html文本。

main.js中给vm实例添加一个数据属性，分别通过v-text和v-html数据绑定，它们之间的区别一目了然

index.html

```
<html>
…
<body>
  <div id="app">
    <p v-text="msg2"></p>
    <p v-html="msg2"></p>
 </div>
</html>
```

main.js

```
…
var vm = new Vue({
  el:"#app",
  data:{
    msg:"欢迎学习vue",
    msg2:"<h1>我是一个H1标题，让我变大！</h1>"
  }
})
```

v-bind属性和v-on事件

v-bind: 元素属性绑定，绑定的数据，在vm实例的data属性中声明
v-on: 元素事件绑定，事件绑定对应的函数，在vm实例的methods属性中声明

```
<!-- v-bind: 指令可以被简写为 :要绑定的属性 -->
<input type="button"  value="按钮" v-bind:title="mytitle"/> -->
<!-- v-bind 中，可以写合法的JS表达式 -->
<input type="button" value="按钮" :title="mytitle+'123'" />

<!-- v-on:事件绑定机制 -->
<input type="button" value="按钮" v-on:click="show1('xx')" />
<!-- v-on: 指令可以被简写为 @要绑定的方法 -->
<input type="button" value="按钮" @click="show" />
```

*: v-on只是提供了便捷的绑定事件方式，并不是改变原生的js事件，所以所有的js事件都是支持的，
https://developer.mozilla.org/zh-CN/docs/Web/Events

v-bind属性和v-on事件

案例:根据自己对v-bind与v-on的学习理解，做一个文字跑马灯效果，提供如下界面代码,完成main.js中的代码部分？

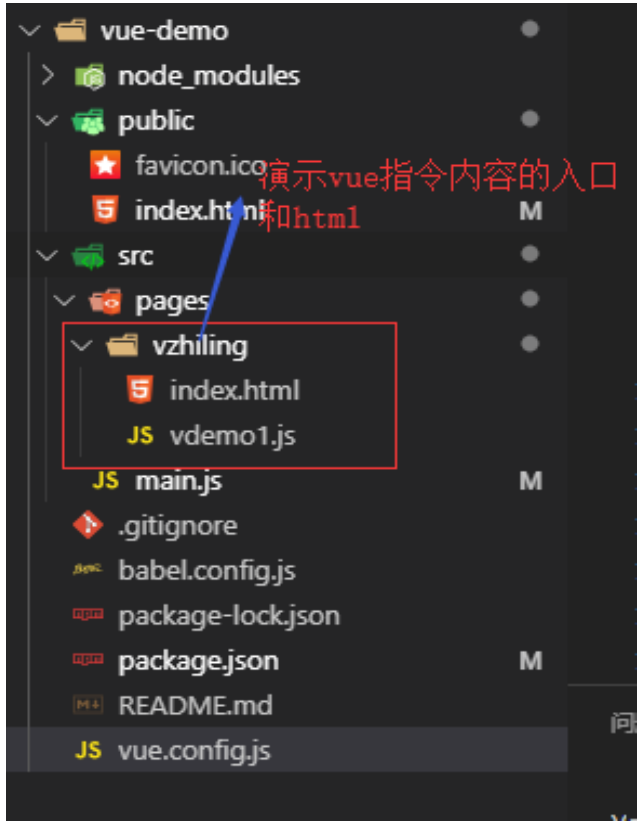
```
<body>
 <div id="app">
    <input type="button" value="启
    <input type="button" value="停
    <h4>{{ msg }}</h4>
 </div>
</body>
```

```
methods:{
  start(){
    if(this.interval!=null){
      return;
    }
    this.interval = setInterval(() => {
      var first=this.msg.substring(0,1);
      var all=this.msg.substring(1);
      this.msg=all+first;
    }, 200);
  },
  stop(){
    clearInterval(this.interval);
    this.interval=null;
  }
}
```

main.js ?

v-bind属性和v-on事件

单页面引用，我们只有一个入口和一个模板文件，每次的代码都需要写在相同的位置，为了相对独立的保留对于vue每一个指令的学习，在vue.confg.js中添加如下配置，每次修改项目的入口和模板文件， 方便保留之前的代码。




```
module.exports={
   chainWebpack: config => {
      // 添加别名
      config.resolve.alias
        .set('vue$', 'vue/dist/vue.js') //修改vue默认导入路径
   },
   pages:{
      index:{
         //入口
         entry: 'src/pages/vzhiling/vdemo1.js',
         //模板来源
         template: 'src/pages/vzhiling/index.html'
      }
   }
}
```

事件修饰符

事件修饰符是**@事件名:事件修饰符=“xx”**，在事件绑定过程中，对于事件不同情景下的调整作用.

修饰符	作用
.stop	阻止事件冒泡，当事件在自身消费后，不再传递事件
. prevent	阻止默认行为，比如a 链接，响应事件后，不再响应默认事件
. capture	捕获事件,事件冒泡之前的事件传递到自身，自身先响应事件，再向下传递
. self	只有事件发生在自身的时候才会响应，冒泡传递过来的事件不做响应
. once	事件在自身只会响应一次，一次后绑定的这个事件失效

设定外层包含内层结构的div, 分别给div设施v-on:click事件并添加表格中修饰符，体会修饰符的作用?

v-model双向数据绑定

v-bind 或者{{xx}} 插值表达式只能实现数据的单向绑定，.从 M 自动绑定到 V 无法实现数据的双向绑定,当v变化的时候，M也自动更新,这个就是双向数据绑定，而v-model就是实现双向数据绑定，但注意的是，v-model只能运用在表单元素中，通常V会改变，也只是应用在表单元素中

```
<div id="app">
    <!-- 注意: v-model 只能运用在 表单元素中 -->
    请说出你的座右铭:<input type="text" style="width:100%;" v-model="msg" @keyup="showMsg" />
</div>
```

vue实例的数据绑定:

```
var vm  = new Vue({
  ..
  data：  {msg:””},
  methods:{
     showMsg(){console.log(this.msg)}
  }
})
```

vue中的样式

vue中要为html的标签元素添加样式提供了2种方式：

- 为元素添加class样式
- 为元素添加style样式

class样式: vue中为元素添加class样式，通过v-bind:class属性绑定的方式完成, 它有主要以下2种方式的支持：

- class属性值直接传递一个数组:
  数组中可以使用三元表达式来控制样式的启用与否，或者数组中用对象的形式代替三元表达式控制样式的启用与否

```
<h1 :class="['thin', 'italic']">这是一个很大很大的H1，大到你无法想象！！！</h1>
<h1 :class="['thin', 'italic', flag?'active':'']">这是一个很大很大的H1，大到你无法想象！！！</h1>
<h1 :class="['thin', 'italic', {'active':flag} ]">这是一个很大很大的H1，大到你无法想象！！！</h1>
```

- class属性值传递data属性的绑定数据

```
<h1 :class="classObj">这是一个很大很大的H1，大到你无法想象！！！</h1>
```

*: classObj以及flag都是vm实例上data对象绑定的属性数据

vue中的样式

style样式: vue中为元素添加style样式，通过v-bind:style属性绑定的方式完成, 它也有主要以下2种方式的支持：

- v-bind绑定style属性，传递一个样式对象

```
<h1 :style="{color: 'red', 'font-size': '40px'}">这是一个善良的H1</h1>
```

- v-bind绑定style属性，传递data属性的绑定数据
  通过属性绑定的形式，可以应用多个样式

```
<h1 :style="styleObj1">这是一个h1</h1>
<h1 :style="[ styleObj1, styleObj2 ]">这是一个h1</h1>
```

```
data: {
      styleObj1: { color: 'red', 'font-weight': 200 },
      styleObj2: { 'font-style': 'italic' }
   }
})
```

v-for和key

v-for 就是循环指令，通过遍历数据，循环元素

- 遍历普通数组

```
<p v-for="(item,index) in list">索引:{{index}}----每一项:{{item}}</p>
```

- 遍历对象数组

```
<p v-for="(item,index) in listObj">索引:{{index}}----每一项id:{{item.id}}:每一项name:{{item.name}}</p>
```

- 遍历对象

```
<p v-for="(val, key, i) in user">  {{key}}:{{ val }} -- 索引:{{i}}</p>
```

- 遍历数字

```
 <p v-for="count in 10">这是第 {{ count }} 次循环</p>
```

*: list、listObj、user都是vm实例的data绑定数据，比如:

```
list: [1,2,3,4,5,6]
listObj:[{id: 1,name: ‘zs1’},{id: 2,name: 'zs2’},{id: 3,name: 'zs3’}]
user:{uid:"007",name:"James Bande",job:"特工",salary:"10000"}
```

v-for和key

官方说明在使用v-for指令的时候:当**在组件上使用 v-for 时，key 现在是必须的**，我们现在并没有在vue组件中使用v-for指令，所以可以不带key属性，但是建议，在任何场景中使用v-for指令，都要带上key属性。比如:当在动态循环li标签时会出现的checkbox问题

```
<div id="app">
    <label>Id:
      <input type="text" v-model="id">
    </label>
    <label>Name:
      <input type="text" v-model="name">
    </label>
    <input type="button" value="添加" @click="add">

    <!-- v-for 循环的时候，key 属性只能使用 number 或者 string,同时要确保key的属性值要是唯一的 -->
    <!-- 如果这里不使用key属性，checkbox选中框会出现对应不上的问题 -->
    <p v-for="item in list" :key="item.id">
      <input type="checkbox">{{item.id}} --- {{item.name}}
    </p>
  </div>
```

v-if和v-show

v-if和v-show都可以控制元素的显示与隐藏,他们的区别在于:

- v-if：每次都会重新删除或创建元素
- v-show:每次不会重新进行DOM的删除和创建操作，只是切换了元素的 display:none 样式

v-if有较高的切换性能消耗，而v-show有较高的初始化渲染消耗，如果元素频繁切换显示与隐藏，建议使用v-show

```
<body>
  <div id="app">
      <input type="button" value="toggle" @click="toggle">
      <h3 v-if="flag">这是用v-if控制的元素</h3>
      <h3 v-show="flag">这是用v-show控制的元素</h3>
   </div>
</body>
```

```
...
    data:{
        flag:true
    },
    methods:{
        toggle(){ this.flag = !this.flag }
    }
```

04 本章总结

本节总结

- Vue实例对象的基本代码结构
- 调整webpack配置方法
- 插值表达式运用
- v-cloak解决插值表达式闪烁问题
- v-text与v-html设置元素内容
- v-bind属性绑定和v-on事件绑定
- v-on事件修饰符
- v-model表单元素双向数据绑定
- vue的样式操作
- v-for指令和key属性
- v-if和v-show

Vue指令综合案例

汽车品牌管理

软通大学
ISOFTSTONE UNIVERSITY

本节目标

- vue-devTools安装
- 案例功能实现

目录 CONTENTS

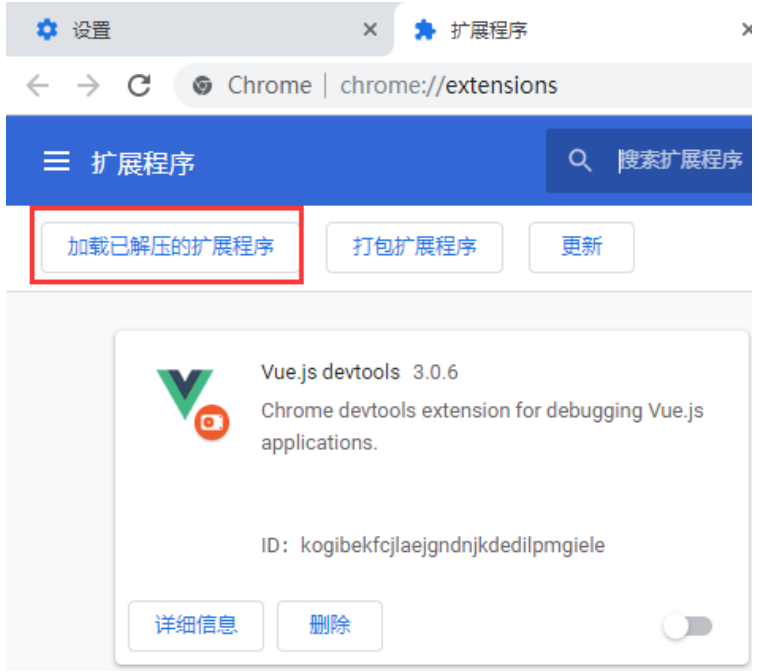
01 vue-devTools安装

vue-devTools安装

Vue.js devtools是基于google chrome浏览器的一款调试vue.js应用的开发者浏览器扩展，可以在浏览器开发者工具下调试代码,安装方式：

- 在线安装(需要翻墙)：chrome商店直接搜索安装
- 离线安装：

  将已经下载解压的vue.js.devTools插件目录手动加载到chrome浏览器的扩展程序中



安装完毕后，F12打开开发者工具,就可以看到Vue插件的选项卡了..

02 案例功能实现

cnpm导入项目依赖库

导入bootstrap，方便快速编写案例的前台界面布局

```
PS D:\vs-work\vue-demo> cnpm install bootstrap -S
√ Installed 1 packages
√ Linked 0 latest versions
√ Run 0 scripts
peerDependencies WARNING bootstrap@* requires a peer of jquery@1.9.1 - 3 but none was installed
peerDependencies WARNING bootstrap@* requires a peer of popper.js@^1.14.7 but none was installed
√ All packages installed (1 packages installed from npm registry, used 3s(network 3s), speed 336.09kB/s, json 1(6.
```

*:凡是在npm 安装依赖的时候出现 上图所示的警告，表示当前这个库依赖的库没有安装上，只要将要求的依赖库安装上即可

```
PS D:\vs-work\vue-demo> cnpm i jquery@1.9.1 -3 -S
√ Installed 1 packages
√ Linked 0 latest versions
√ Run 0 scripts
√ All packages installed (1 packages installed from npm registry, used 2s(network 2s), speed 210.95kB/s,
PS D:\vs-work\vue-demo> cnpm i popper.js@1.14.7 -S
√ Installed 1 packages
√ Linked 0 latest versions
√ Run 0 scripts
√ All packages installed (1 packages installed from npm registry, used 2s(network 2s), speed 265.46kB/s,
```

cnpm导入项目依赖库

cnpm常用安装命令的区别:

- npm i module_name -S = > npm install module_name --save 写入到 dependencies 对象
- npm i module_name -D => npm install module_name --save-dev 写入到 devDependencies 对象
- npm i module_name -g 全局安装
- npm i module_name 安装到当前目录的nodu_modules,但不会写入到dependencies和devDependencies中，在项目开发时，不建议使用

注意：vscode安装bootstrap插件，在编写bootstrap组件元素时可以给出智能提示，提高效率

添加汽车品牌

html中输入bs3 table即可快速创建bootstrap表格元素，输入bs3 panel 可以快速创建简单面板: 提供输入汽车品牌和名称，点击添加数据。

*:html要用到bootstrap的样式，在入口js文件中，要导入bootstrap的样式

```
import Bootstrap from "bootstrap/dist/css/bootstrap.min.css"
```

删除汽车品牌

table的tr行中，最后一列td, a 标签点击删除，v-on:click事件绑定，删除对应的记录

- a链接需要阻止默认事件：事件修饰符
- 对应methods事件处理方法，使用some循环数组，删除元素

```
...
del(id) {
      //some循环：some循环和forEach不同的是，some循环内可以return true终止循环
      this.brandlist.some((element, index) => {
        if(element.id == id){
          //删除元素
          this.brandlist.splice(index,1);
          return true;
        }
      })
    }
```

查询汽车品牌(过滤)

v-for 循环每一行数据的时候，不再直接 `item in list`，而是 `in` 一个过滤的methods 方法，methods方法绑定属性keywords,其内部再用for循环过滤符合条件数据，返回新的数组

```
<tr v-for="(item,index) in search(keywords)" :key="item.id">
            <td>{{item.id}}</td>
            <td>{{item.name}}</td>
            <td>{{item.ctime}}</td>
            <td><a href="" @click.prevent="del(item.id)">删除</a></td>
</tr>
```

```
...
search(name){
    //filter方法过滤数组：return true,表示这个item符合，数组中保留，return false则相反
    return this.brandlist.filter((element)=>{
        return element.name.includes(name);
    })
 }
```

*: 特别需要注意的是：search函数一定要用return返回结果

本节总结

- Vue指令的综合案例练习

本节练习

- 课堂案例：汽车品牌管理的增加、删除、查询功能 ？

软通大学
ISOFTSTONE UNIVERSITY