
Vue指令
软通大学
ISOFTSTONE UNIVERSITY

本节目标

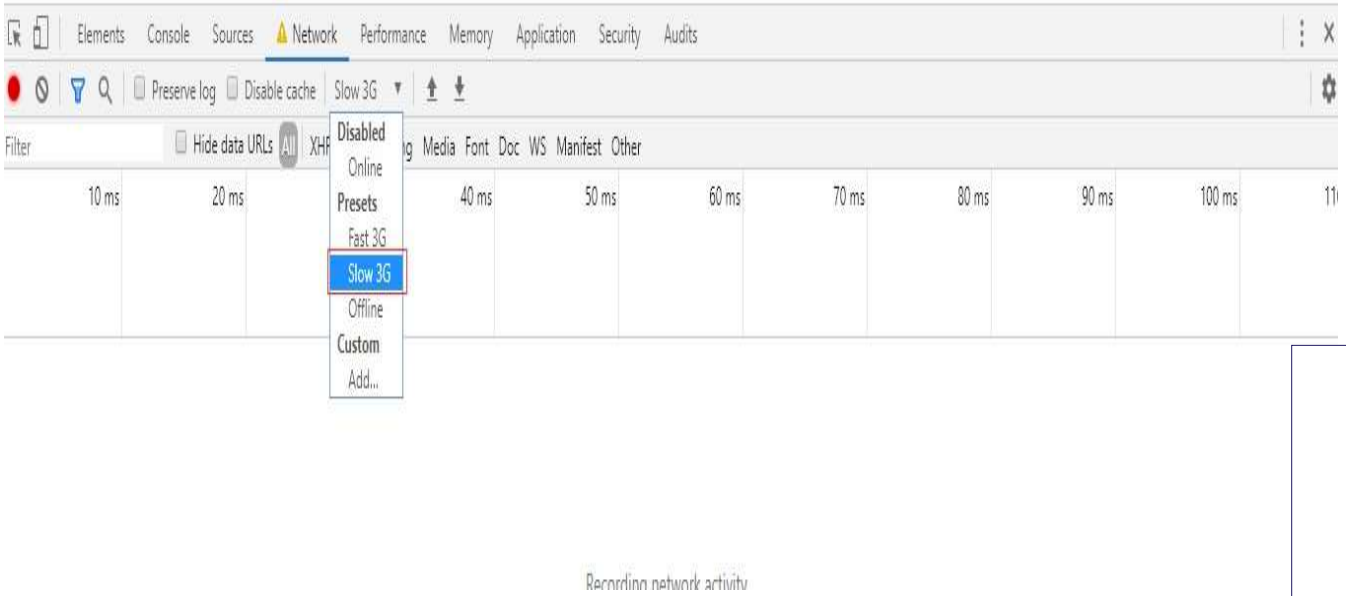
- 掌握全部vue指令的使用

目录 CONTENTS

01v-cloak解决闪烁问题

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

02 v-text与v-html

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>"
  }
})
```

03 v-bind属性与v-on事件

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

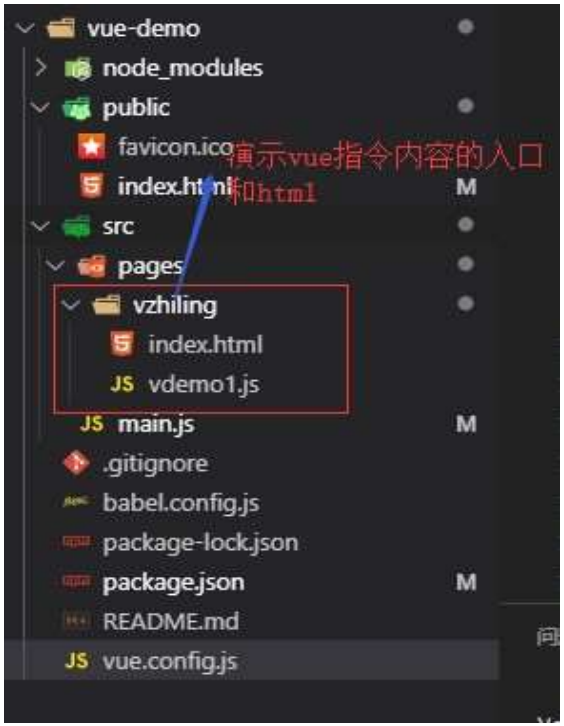
案例:根据自己对v-bind与v-on的学习理解，做一个文字跑马灯效果，提供如下界面代码,完成main.js中的代码部分？

```
<body>
 <div id="app">
    <input type="button" value="浪起来" @click="lang">
    <input type="button" value="低调" @click="stop">
    <h4>{{ msg }}</h4>
 </div>
</body>
```

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'
    }
  }
}
```

04 事件修饰符

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?

05 v-model双向数据绑定

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)}
  }
})
```

06 vue中的样式

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' }
   }
})
```

07 v-for和key

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>
```

08 v-if和v-show

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 }
   }
```

本节总结

- 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

本节练习

- 课堂案例：文字跑马灯效果？

软通大学
ISOFTSTONE UNIVERSITY