# 一、MVVM

是一种软件架构的设计模式,VUE是mvvm的实现者，微信小程序也是。

M model（数据层）

V view（视图层）

**VM viewmodel**

MVVM的核心就是后面的这个VM，就是通过vm，对model层进行监听，**对view进行数据的绑定**，这种效果就可以达到：

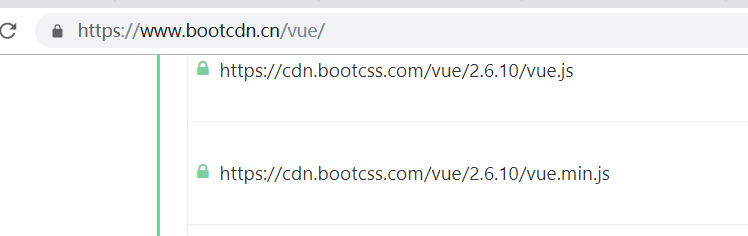
当数据发生改变，就会实时的影响view层的展示的内容；

当view层绑定的数据发生改变，就会实时的影响model层中的数据。

# 二、快速开始

## 1.在页面中导入vue的cdn的js文件。

从该网站获取vue的cdn文件



注意： 开发环境下 使用vue.js， 生产环境下使用 vue.min.js 这样的文件更小。因为vue.min.js中的内容没有格式，没有换行，所以更小。

## 2.在页面中创建一个id是app的div标签

div id="app"

## 3.在script中创建一个vue对象，并且在vue对象中添加内容

vue的内容如下：

|  |
| --- |
| //el: element 指明页面当中的哪个元素会被当前vue对象绑定.  //data: 提供数据  new Vue({    el:'#app',  data:{  name:'zs',  age:20,  gender:'男',  school:{  name:'千锋',  addr:'杭州'  }  },  methods:{  show:function(){  alert("hello vue");  }  }  }); |

## 4.在页面中使用vue对象中的内容

js的作用： 动态改变 页面的内容和页面的样式

页面的内容是两个部分： 页面的数据，页面的函数

通过插值表达式

|  |
| --- |
| <div id="app">  {{name}}  {{show()}}    </div> |

注意：

**插值表达式只能写在被vue对象绑定的那个div中**。

# 三、插（差）值表达式

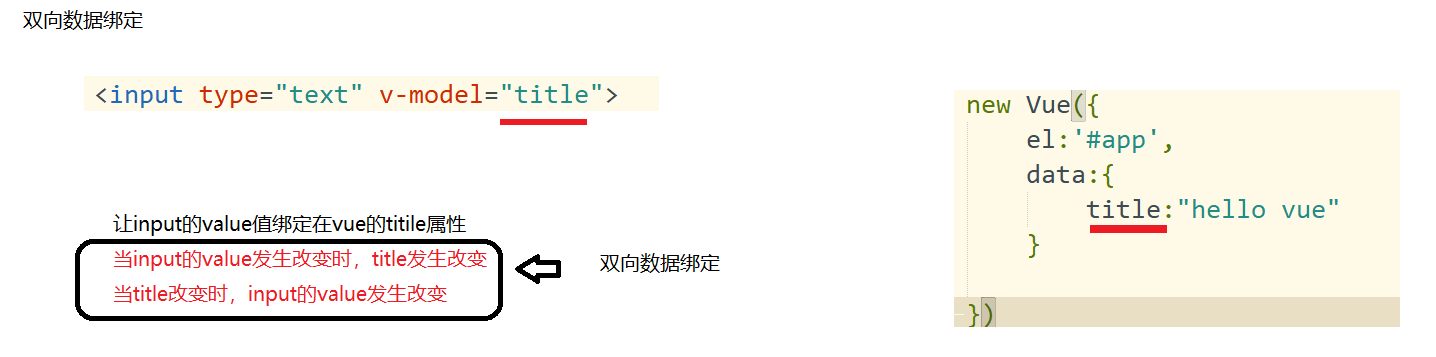
可以通过插值表达式在页面当中展现VUE中的内容（属性、方法）

|  |
| --- |
| 获取vue对象中的数据  {{name}}  获取数组中的某个元素  {{[1,2,3,4,5][2]}}  获取对象中的某个属性  {{ {username:'zs',age:22}.username }}  进行三元运算（本质上还是得到一个值）  {{ 1>0?'是':'否' }}  调用vue对象中的方法  {{show()}}  能够进行简单的运算  {{1000\*2+100/8+20}} |

# 四、Vue中的关键字

## 1.v-model（用在标签后面，绑定一个vue属性）

实现双向数据绑定



## 2.v-on

**事件绑定（绑定一个事件，让事件等于一个函数，通常是按钮标签，可缩写为@事件=“函数名”）**

让html中的元素绑定事件的响应行为，该响应行为在vue对象中定义。

v-on:事件="响应行为"

v-on:click = "btnfn"

还可以简写： @click="btnfn"

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8" />  <title></title>  </head>  <body>  <div id="app">  <button type="button" @click="btnfn">点我</button>  <button type="button" v-on:click="btnfn">点我</button>  </div>  </body>  <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  <script type="text/javascript">  new Vue({  el:'#app',  data:{  title:"hello vue"  },  methods:{  btnfn:function(){  alert("晚上约吗？");  }  }    })  </script>  </html> |

## 3.v-bind（标签的属性等于vue对象中的属性，这是需要用v-bind绑定，可写为 ：属性=“vue对象属性”）

属性绑定

**html标签的中属性可以获取vue对象中的值。**

<a v-bind:href="link">点我</a><br/>

<a :href="link">点我</a>

## 4.v-once（直接在标签中加入v-once即可）

让数据只显示初始化时的值，之后不变。

<input type="text" v-model="title"/>

<p v-once>

{{title}}

</p>

## 5.v-html（用法和v-bind相似，网址作为属性被写在vue中）

将vue中的内容作为html本身的内容

<p v-html="mylink"></p>

mylink:'<a href="http://baidu.com">baidu</a>'

此时出现a标签

## 6.v-text（用法同v-html）

将vue中的内容作为纯文本

<p v-text="mylink"></p>

mylink:'<a href="http://baidu.com">baidu</a>'

此时出现纯文本。

# 五、事件

事件绑定

参数传递

**事件修饰符**

**event.stop 组织当前事件**

**keyup.space 当空格键弹起就会调用**

**keyup.enter 当回车键弹起就会调用**

# 六、计算属性

**计算属性是vue提高运算效率的一种方式，将函数缓存起来，函数的结果存入到浏览器的缓存中，下次调用该函数时，直接使用缓存中的结果。**

**计算属性实际上是一个属性。但是在定义时会定义成方法！**

|  |
| --- |
| **new vue({**  **el:'',**  **data:{}, //属性**  **methods:{}, //方法**  **computed:{**  **getDate:function(){}//属性**  **} //属性**  **})** |

**七、watch**

vue中的watch用于监听（观察）属性，当属性一旦变化，就会调用watch中定义的函数，函数能够得到变化前和变化后的值。

|  |
| --- |
| <body>  <div id="app">  {{count}}  <button type="button" @click="increase">点我</button>  </div>  </body>  <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>  <script type="text/javascript">  new Vue({  el:'#app',  data:{  count:0  },  methods:{  increase:function(){  this.count++;  }  },  watch:{  count:function(newValue,oldValue){  console.log(newValue+"-"+oldValue);  }  }  })  </script> |

# 七、Vue改变页面的样式

1.方式一：

**通过给页面的属性绑定布尔值，来决定元素的样式**

:class="{样式名:来自于vue的布尔值}"

|  |
| --- |
| <div class="demo" :class="{red:flag}">  111  </div>  new Vue({    el:'#app',  data:{  flag:false  },  methods:{    }  }); |

2.方式二：

**在computed中提供class需要的{}对象,用一个函数来控制标签的属性**

|  |
| --- |
| <div class="demo" :class="getStyle">  333  </div>  computed:{  getStyle:function(){  return {  red:this.flag  }  }  } |

方式三：

**使用双向数据绑定v-model**

|  |
| --- |
| .red{  background-color: red;  }  .green{  background-color: green;  }  .yellow{  background-color: yellow;  }  <div class="demo" :class="color">  444  </div>      <input type="text" v-model="color"/>  data:{  flag:false,  color:''  } |

方式四：

绑定多个样式(通过style标签来控制样式)

|  |
| --- |
| <div class="demo" :class="[color,t]">  555  </div>  data:{  flag:false,  color:'',  t:'textColor'  },  .yellow{  background-color: yellow;  }  .textColor{  color:pink  } |

方式五：

通过style属性来改变样式

|  |
| --- |
| <div class="demo" :style="[mystyle,{height:h+'px'},{width:w+'px'}]">  666  </div>  new Vue({    el:'#app',  data:{  flag:false,  color:'',  t:'textColor',  h:20,  w:30  },  methods:{    },  computed:{  getStyle:function(){  return {  red:this.flag  }  },  mystyle:function(){  return {  backgroundColor:this.color,    }  }  }  }); |

# 八、v-if分支语句

|  |
| --- |
| <body>  <div id="app">  <p v-if="flag">  今天天气很舒服！  </p>  <p v-else-if="rich">  今天天气很燥热！晚上要去放松一下！  </p>  <p v-else="rich">  晚上只能自嗨！  </p>    </div>    </body>  <script src="https://cdn.bootcss.com/vue/2.6.10/vue.js"></script>  <script>    new Vue({    el:'#app',  data:{  flag:false,  rich:false  },  methods:{    }  });  </script> |

v-show和v-if的区别

v-show是让元素的display属性产生变化： none/block

v-if是让元素出现在dom中，或者不出现

因此，对于频繁变化的元素来说，建议使用v-show，效率更高！