# 1.data函数p158

|  |
| --- |
|  |

# 2.数据绑定p159

|  |
| --- |
|  |
|  |

# 3.vue属性绑定p160

## 因为vue的{{ }}不能够使用在元素的属性上面,所以如果你需要动态改变属性,就需要使用属性绑定

|  |
| --- |
|  |

### 注意:可以简写: v-bind:href == :href

# 4.vue事件处理-vue事件绑定p162

|  |
| --- |
|  |
|  |

## 事件绑定的指令是v-on:事件名=”事件处理函数” 可以简写为@事件名 = ”事件处理函数”,这个事件处理函数一定要在methods里面定义.多个函数之间用逗号隔开.

# 5.vue事件处理-vue事件对象p163

|  |
| --- |
|  |

## 在vue中是这样子获取事件对象的: 如果调用事件处理函数没有传递参数,函数的形参就代表事件对象.如果你调用事件处理函数传递了参数,那么形参就代表实参(此时形参不是事件对象),也就是你传递的参数.但是如果我们传递的实参是一个$event,它就表示事件对象.也可以在js代码里面直接使用event来获取.不过其实这种方式已经不推荐使用了.

|  |
| --- |
|  |

# 6.vue事件处理-vue事件修饰符p164

|  |
| --- |
|  |
|  |
|  |

## 原生的dom也提供了一些方法来处理事件流:

|  |
| --- |
|  |

# 演练1.数据绑定:

## 1>我们用模板来创建一个lesson3项目(需要进入项目目录运行npm install)

|  |
| --- |
|  |

### 然后我们打开App.vue添加下面的代码,用快捷键vdata,然后修改一下,就会变为下面的样子

|  |
| --- |
|  |

## 2>然后我们在div标签里面使用{{}} 在里面写message这样子就可以把message的内容显示在页面上

|  |
| --- |
|  |

## 3>接下来,我们来演练运行v-text和v-html,我们先修改一下data函数的内容

|  |
| --- |
|  |

## 4>然后我们来修改模板的内容

|  |
| --- |
|  |

### 运行程序,效果如下

|  |
| --- |
|  |

### 可见,v-text特性不解析html标签,v-html特性会解析html标签,他们是单向绑定的.{{}}也是单向的.{{}}里面的标签也不会解析

# 演练2.属性绑定

## 1.用模板来创建一个lesson4项目,然后进入这个项目,执行npm install 安装依赖

|  |
| --- |
|  |

## 2.打开App.vue,在里面添加一些数据(快捷键vdata)

|  |
| --- |
|  |

## 3.然后我们在模板里面使用这些数据这里使用了属性绑定的方法,

|  |
| --- |
|  |

### 运行程序,效果如下

|  |
| --- |
|  |

## 4.然后我们可以给他添加一点样式,使得页面稍微好看一点点

|  |
| --- |
|  |

### 运行程序,效果如下

|  |
| --- |
|  |

# 演练3.vue事件绑定

## 1,新建一个lesson5项目,安装依赖然后打开App.vue,在script标签里面把data函数的数据改为counter,然后在的data函数下面添加一个methods属性,快捷键vmethod,然后添加2个方法一个增加值一个减少值

|  |
| --- |
|  |

## 2.然后修改一先template里面的内容如下,也就是进行事件绑定

|  |
| --- |
|  |

### 运行程序,效果如下

|  |
| --- |
|  |
|  |
|  |

# 演练4.事件修饰符

## 1>用模板新建一个项目,起名lesson7-xxx,然后进入App.vue,在模板了吗添加3个div,class分别叫做outer,middle和inner

|  |
| --- |
|  |

## 2>然后,给他们添加一些样式

|  |
| --- |
|  |

## 3>然后我们先在methods里面创建一些方法,可以先写定义方法,以后再实现方法

|  |
| --- |
|  |

## 4>给他们用vue指令绑定事件

|  |
| --- |
|  |

## 5>我们给那几个方法添加一些简单实现代码

|  |
| --- |
|  |

### 运行程序,点击inner对应的div,你会发现3个函数都调用了.

|  |
| --- |
|  |

### 然后我们把控制台清理一下,点击中间的div,发现事件2和事件3会触发,应该也是事件冒泡

|  |
| --- |
|  |

## 6>需要阻止事件冒泡,其实也很简单,可以使用DOM事件对象的stopPropagation方法,我们在fun3函数里面添加这一行代码

|  |
| --- |
|  |

### 然后我们再次运行程序,把控制台的清理一下,然后点击最里面的div,效果如下

|  |
| --- |
|  |

## 7>我们在模板中添加应该超链接

|  |
| --- |
|  |

## 8>我们在methods里面添加一个fun4函数

|  |
| --- |
|  |

## 9>然后我们给超链接的点击事件绑定这个函数

|  |
| --- |
|  |

### 然后运行程序,在窗口中点击超链接,就会弹出一个警告框,然后链接不会跳转到百度

|  |
| --- |
|  |

### 注意:虽然能够工作,但是这样子是不推荐的,我们需要使用vue提供的事件修饰符,注意不是函数,是.xxx形式的修饰符.此时就不需要在方法里面调用dom默认的事件流方法了.

## 10>我们把方法里面的调用dom默认的事件流方法的代码注释掉.

|  |
| --- |
|  |

## 11>然后在@click后面添加.prevent=”函数名称”,即使函数里面有代码,但是添加了prevent属性后,代码不会执行,在vue里面.stop = dom事件对象.stopPropagation,.prevent= dom事件对象.preventDefault

|  |
| --- |
|  |

## 然后运行程序,发现,事件不会冒泡而且超链接的默认跳转行为也没有发生

|  |
| --- |
|  |
|  |