# VUE的基本使用day1

## vue基础入门

### 什么是vue：

●vue是一套同于构建用户界面（用vue往html页面中填充数据）的前端框架（现成的解决方案，程序员只能遵循框架的规范，去编写自己的业务功能）

●学习vue就是学习vue框架中规定的用法，例如vue中的指令、组件（对UI结构的复用）、路由、vuex。

●只有把上面罗列的内容掌握后才有开发vue项目的能力

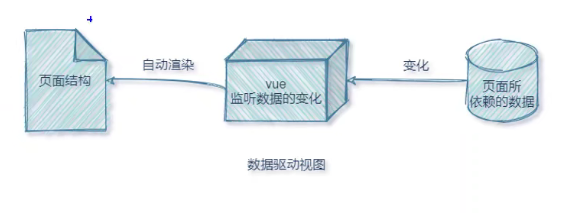
### vue的特性

Vue框架的特性，主要体现再如下两方面：

1. 数据驱动视图
2. 双向数据绑定

#### 数据驱动视图

在使用了vue的页面中，vuehi监听数据的变化，从而子自动重新渲染页面的结构。示意图如下：

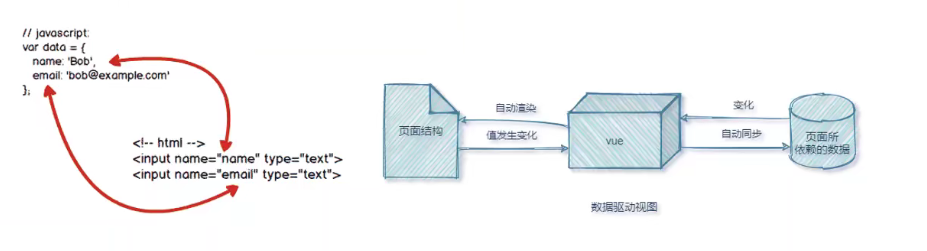


好处：当页面数据发生变化时，页面会自动重新渲染，程序员只需维护好数据，页面结构会被vue自动渲染出来

注意：数据驱动视图是单向的数据绑定

#### 双向数据绑定

在填写表单时，双向数据绑定可以辅助开发者在不操作DOM的前提下，自动把用户填写的内容同步到数据源中，示意图如下：



在网页中，from表单负责采集数据，ajax负责提交数据

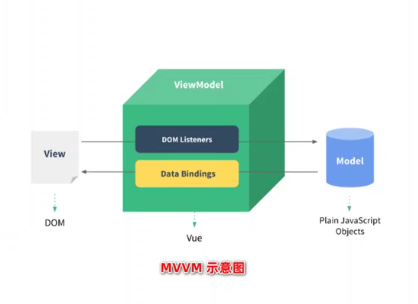
js数据的变化，会被自动渲染到页面上

页面上表单采集的数据发生变化的时候会被vue自动获取到并更新到js数据当中

好处：开发者不需要手动操作DOM元素来获取表单元素最新的值

### MVVM

MVVM是vue实现数据驱动视图和双向数据绑定的核心原理。MVVM是指Model、View和ViewModel，它把每个HTML页面都拆分成了这三个部分，如图所示：



在MVVM概念中：

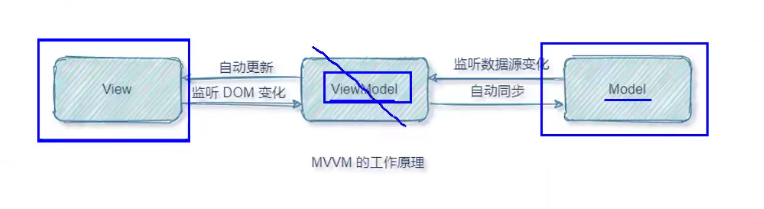
Model表示当前页面渲染时所依赖的数据源

View表示当前页面所渲染的DOM结构

ViewModel表示Vue的实例，它是MVVM的核心

MVVM的工作原理

ViewModel作为MVVM的核心，是它把当前页面的数据源（model）和页面的结构（view）连接在了一起。



当数据源发生变化时，会被ViewModel监听到，VM会根据最新的数据源自动更新页面的结构

当表单元素的值发生变化的时候，也会被VM监听到，VM会把变化后最新的值自动同步到Model数据源中

注意：数据驱动视图和双向数据绑定的底层原理是MVVM（Model数据源、View页面结构/视图、ViewModel vue的实例对象）

### Vue基本使用

#### 1.vue基本代码

1. 导入vue.js的script脚本文件
2. 在页面中声明一个将要被vue所控制的DOM区域
3. 创建vm实例对象（vue实例对象）



基本代码与MVVM的对应关系：



#### 2.vue调试工具和数据数据驱动视图

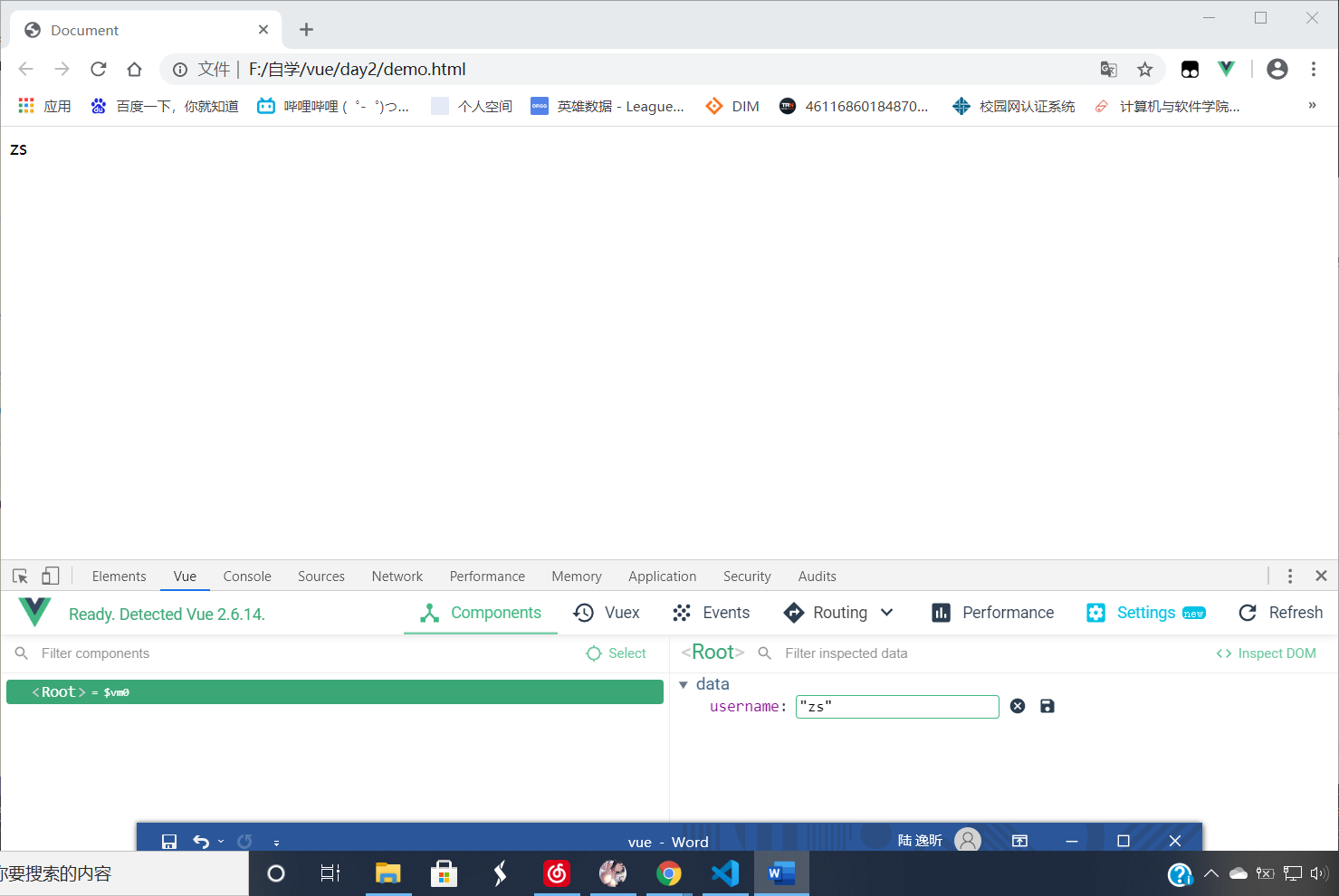
安装：



配置：



使用：



f12打开控制面板里的vue可进行数据的操作

#### 3.指令

指令是vue为开发者提供的模板语法，用于辅助开发者渲染页面的基本结构

vue中指令按照不同的用途可以分为以下6大类：

① 内容渲染指令

② 属性绑定指令

③ 事件绑定指令

④ 双向绑定指令

⑤ 条件渲染指令

⑥ 列表渲染指令

指令是vue开发中最基础、最常用、最简单的知识点

注意：vue指令在使用时只对el属性选择器选取的区域内生效，区域外不生效

##### 内容渲染指令

内容渲染指令用来辅助开发者渲染DOM元素的文本内容。常用的内容渲染指令有如下3个：

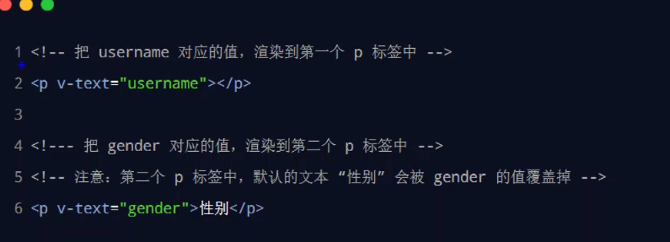
● v-text

● {{ }}

● v-html

**v-text：**

用法示例：

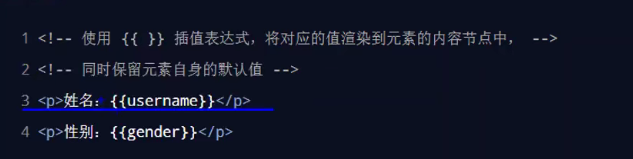


v-text缺点：会覆盖DOM元素内原有的内容

**{{ }}语法：**

vue提供的{{ }}语法，专门用来解决v-text会覆盖默认文本内容的问题。这种{{ }}语法的专业名称为插值表达式

用法：



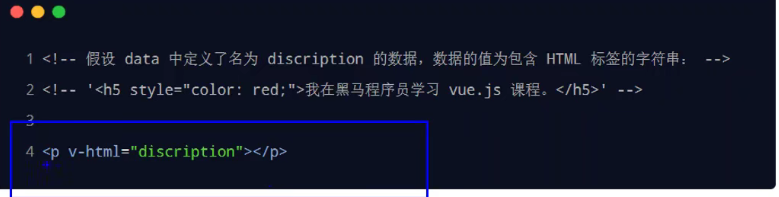
插值表达式在实际中使用最多，只是内容的占位符，不会覆盖原有的内容

插值表达式只能用在内容节点不能用在属性节点，例：<input type=”{{typr}}”>这样是不行的

**v-html：**

v-text指令和插值表达式只能渲染纯文本内容。如果要把包含HTML标签的字符串渲染为页面的HTML元素，则需要用到v-html

用法：



##### **属性绑定指令**

如果需要为DOM元素的属性动态绑定属性值，则需要用到v-bind属性绑定指令。用法示例如下：

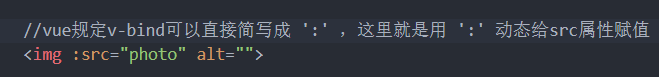


在需要动态赋值的元素前使用v-bind



动态的值

vue规定v-bind可以简写成 ‘ : ’ 例：



##### 在插值和属性绑定中编写JS语句

在vue提供的模板渲染语法中，除了支持绑定简单的数据值之外，还支持Javascript表达式的运算，例如：

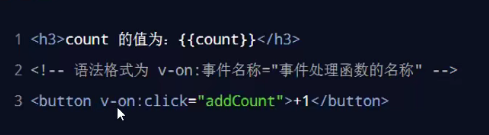


##### 事件绑定指令

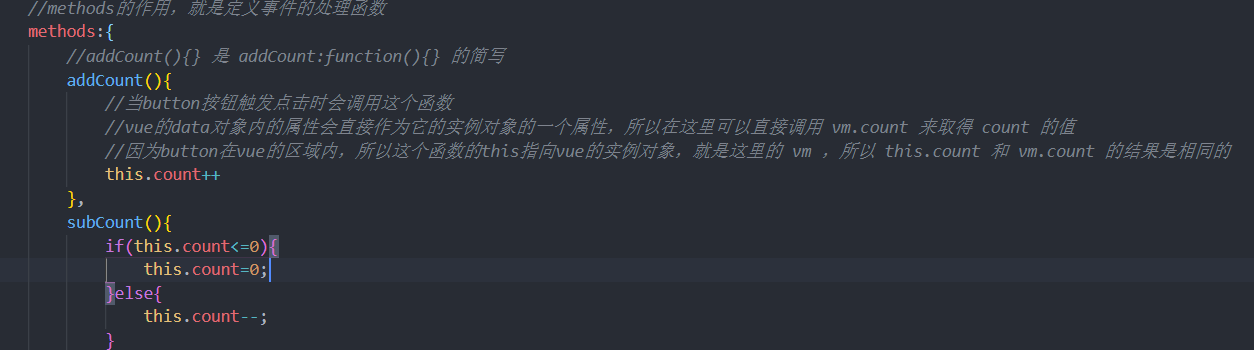
vue提供了v-on事件绑定指令，用来辅助程序员为DOM元素绑定事件监听。

语法格式：

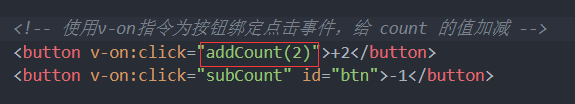
<DOM元素 v-on:事件名称=”调用的事件处理函数的名称”>

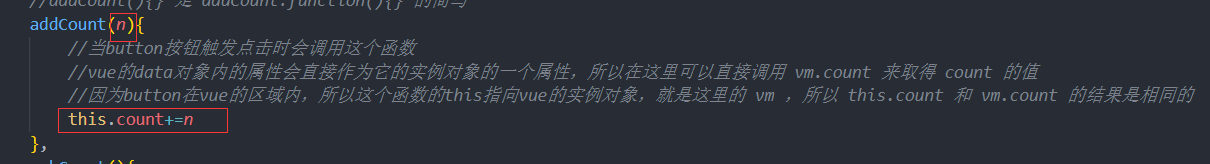


在vue的实例对象中使用methods来进行事件处理函数的配置：



在触发事件的时候由于是调用methods中定义的处理函数，所以可以用（）进行传参：





vue规定v-on事件绑定指令可以简写成@



注意：原生DOM对象中有onclick、oninput、onkeyup等原生事件，替换为vue的事件绑定形式后分别为：v-on：click 、v-on：input 、v-on：keyup 等等

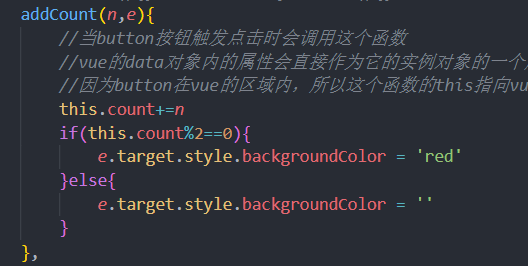
在调用事件处理函数时不传参数，则默认携带一个参数e，e内存储了关于事件源的信息，例如: e.target 可以拿到事件源是谁

例：根据conut的值修改button的颜色



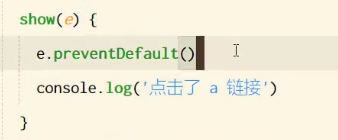
如传递了参数，则vue提供了一个内置变量，名字叫：$event 它就是原生的DOM事件对象e，要使用时需要将$event也作为参数传递过去，注意传参的位置要对应





##### 事件修饰符

在事件处理函数中调用event.preventDefault()（阻止默认行为）例：



或event.stopPropagation()（阻止冒泡行为）是非常常见的需求。因此，vue提供了事件修饰符的概念，来辅助程序员更方便的对事件的触发进行控制。常用的5个事件修饰符如下：（前两个常用）

.prevent : 阻止默认行为（例如：阻止a链接的跳转、阻止表单的提交等）

.stop : 阻止事件冒泡

.capture : 以捕获模式触发当前的事件处理函数

.once : 绑定的事件只触发1次

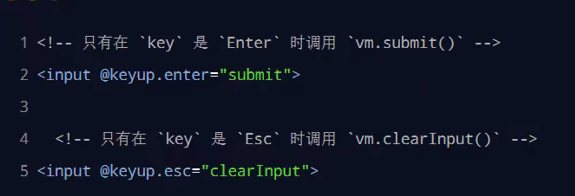
.self : 只有在event.target是当前元素自身时触发事件处理函数

使用示例：



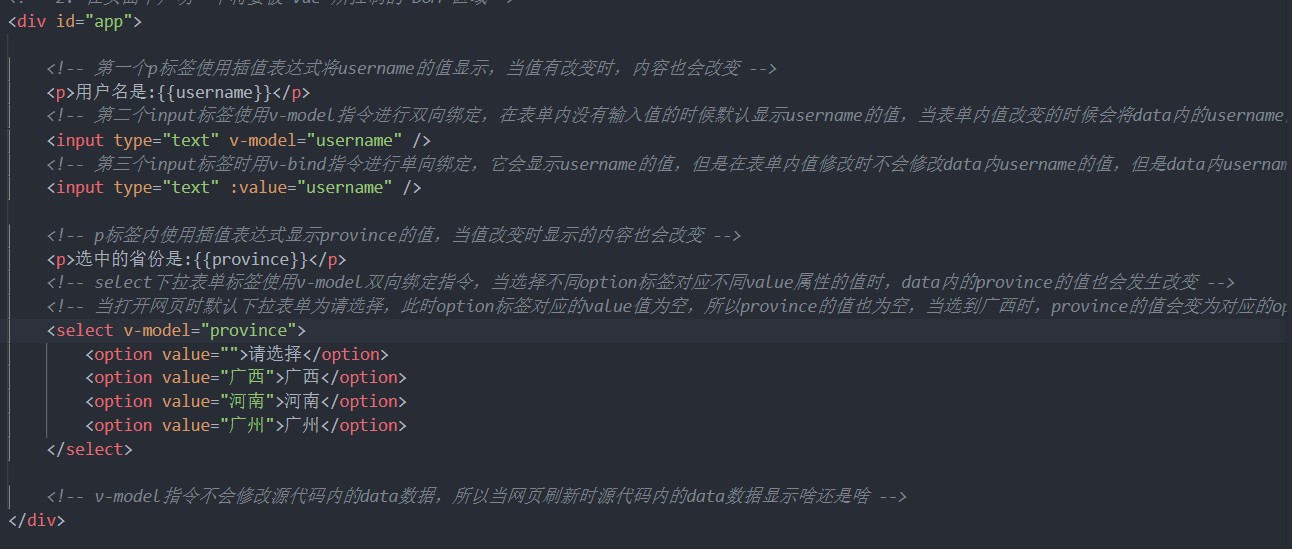
##### 按键修饰符

在监听键盘事件的时候往往我们需要判断详细的按键。此时，可以为键盘相关的事件添加按键修饰符，例如：



##### 双向绑定指令

vue提供了v-model双向数据绑定指令，用来辅助开发者在不操作DOM的前提下，快速获取表单元素，例：





即使用v-model指令绑定的表单，它的value的值就与v-model绑定的值同步

双向数据绑定即当有一边数据发生变动时两边的数据都会发生变动，这是双向（v-model）

单向数据绑定即只有规定好的一边数据发生改变时才会改变两边的数据，没规定的那边数据即使修改也不会改变另外一边的数据（v-bind）

注：只有表单元素才能使用v-model指令

例如：1.input输入框（type=“radio”、type=“checkbox”type=“xxx”等等属性）

2.textarea

3.select（下拉表单）

使用v-model时不用设定input标签的value属性了，v-model指令默认监听了value属性会对其进行修改

##### v-model修饰符

为方便对用户输入的内容进行处理，vue为v-model指令提供了3个修饰符，分别是：

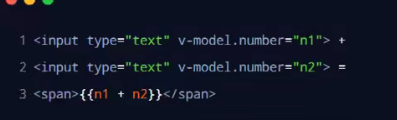
.number : 自动将用户的输入值为数值类型，例：<input v-model.number=”age” />

.trim : 自动过滤用户输入的首位空白字符，例：<input v-model.trim=”msg” />

.lazy : 在“change”时而非“input”时更新，相当于将双向绑定更改为单向绑定，例：

<input v-model.lazy=”msg” />

具体实例：



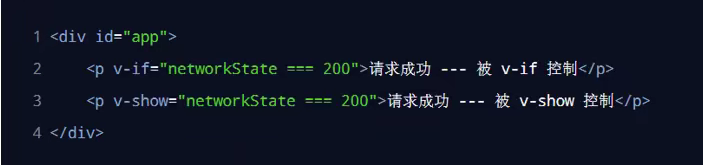
##### 条件渲染指令

条件渲染指令用来辅助开发者按需求控制DOM的显示与隐藏。条件渲染指令有如下两个：

●v-if

●v-show

实例用法如下：



这两个指令当判断条件为false时都会隐藏

区别：

v-if指令会动态移除该标签

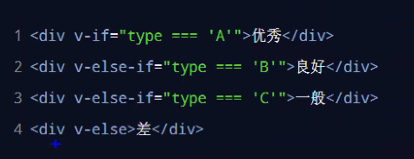
○如果刚进入页面的时候，某些元素默认不需要被展示，而且后期这个元素也不需要频繁的展现出来的时候，使用v-if指令性能更好，不过现在的电脑性能不太需要这种优化，直接使用v-if即可

v-show指令会给该标签添加样式 display:none 使其隐藏

○如果要频繁的显示和隐藏样式就使用v-show性能更好

v-if扩展：v-else和v-else-if

v-else和v-else-if指令，顾名思义，充当v-if的“else-if”块，可以连续使用，例：



注意：v-else和v-else-if指令必须配合v-if指令一起使用，否则它将不会被识别

##### 列表渲染指令

vue提供了v-for列表渲染指令，用来辅助开发者基于一个数组来循环渲染一个列表结构。v-for指令需要使用item in items 形式的特殊语法，其中：

●items 是待循环数组

●item 是被循环的每一项



给那个标签循环就给哪个标签加v-for，例如想给表单动态生成行就给tr加v-for

v-for指令还支持一个可选的第二个参数，即当前项的索引。语法格式为

（item，index）in items，示例如下：



注意：v-for指令中的item项和index索引都是形参，可以根据需要进行重新命名，如：

（user，i）in userlist

vue官方建议，当使用了v-for指令就在调用指令末尾处加 :key 属性；而且尽量把数组中的id作为key属性的值，因为官方对于key的属性是有要求的，只能为数字型或字符串型；同时key的值不能重复，当key的值重复的时候终端会报错

具体使用例：

