1. 基础

(1)Vue是MVVM框架, 数据的双向绑定

M -- Model(模型, 后端传递的数据) --> data

V -- View(视图, 所看到的页面) -->html

VM -- ViewModel(视图模型, 链接M和V的桥梁) --> new Vue

(2)实例参数(new Vue)

## el: “id” 实例所控制的html区域

## data: {} 存放的数据( 子组件必须是一个方法 return {} )

## methods: {} 存放方法

(3)模板语法

{{ }}

1. 指令

(1)v-text：展示其对应的文本

(2)v-html：把值中的标签渲染出来

(3)v-on：事件绑定

### 事件修饰符：

+ .stop 阻止冒泡

+ .prevent 阻止默认事件

+ .capture 添加事件侦听器时使用事件捕获模式

+ .self 只当事件在该元素本身（比如不是子元素）触发时触发回调

+ .once 事件只触发一次

(4)v-bind：适用于绑定行内属性

## v-bind:style

## v-bind:class

(5)v-for：循环显示元素的;可以循环数组, 也可以循环对象、数字、字符串

(注意必须绑定绑定key值, 遍历数组对象时需要用item.key(对象中的键)来访问值)

v-for有四种用法 数组， 数组带对象， 对象， 对象带数组。数字(数字是从1开始) 🡪v-for=”item in 10”十次1到10

##实例

<div v-for="(val, key, i) in userInfo">{{val}} --- {{key}} --- {{i}}</div>

(6)v-model：双向绑定

(7)v-show=‘布尔’：布尔为 true 显示，false 隐藏。

(8)v-if=‘布尔’：布尔为 true 显示，false 隐藏。 (惰性)

##v-show和v-if总结

一般来说，v-if 有更高的切换消耗而 v-show 有更高的初始渲染消耗。因此，如果需要频繁切换 v-show 较好，如果在运行时条件不大可能改变 v-if 较好。

(7)v-once：对应的标签只渲染一次

(8)v-cloak：防止页面加载时出现 vuejs 的变量名指令，在项目在开发中，经常会遇到当数据加载时vue的一些标签就会闪现，然后等数据加载完后消失，这时候就需要用到 v-cloak 来解决。

1. 过滤器(filter)

全局定义过滤器-🡪



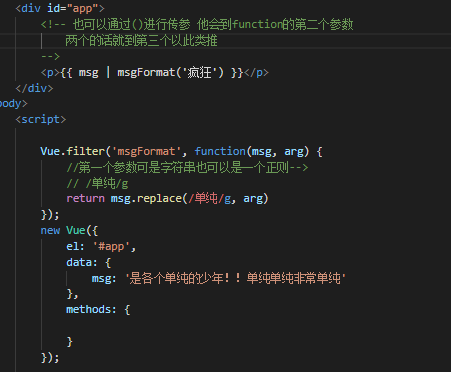
注意：过滤器只用用在mustache插件表达式( {{}} )和v-bind中。

参数可以填data也可以填data里面的message等等(填需要过滤的字符)。

在html中表现形式为-🡪



过滤器传参的使用🡪

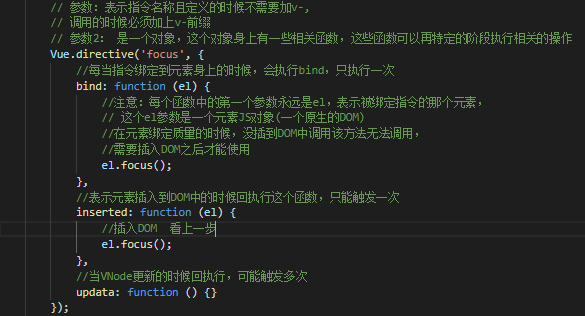


可以多个过滤器处理 他会从左到右依次使用过滤器-🡪



先msgFormat过滤在test过滤

1. 自定义指令



注意：设置样式指令的时候不需要inserted。因为不插入DOM就能生效。(只需要绑定指令到元素身上就能生效)

总结：样式相关的在bind中设置。 行为有关的在inserted设置

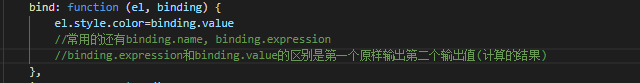
自定义指令传参(设置第二个参数)

Html中



注意：”yellow”表示一个变量会到data里面去找， “’yellow’”表示一个字符，传参

Script中



简写自定义指令

只需要bind和updata不需要inserted可简写成🡪

Vue.directive(‘自定义指令名称’, function () {})---》



如果自定义有单位的可以paseInt(binding.value) + ‘单位’ 来容错。

1. 生命周期行数

生命周期钩子 = 生命周期函数 = 生命周期事件

1. beforeCreate在beforeCreate生命周期函数执行的时候，

data和methods中的数据没有初始化。

1. created在created生命函数执行的时候，data和methods都已经初始化好了。表示如果要调用methods中的方法或者要操作data中的数据最早只能在created中操作

data和methods初始化完成

1. 接下Vue开始编译模板，把vue中的指令执行，最终在内存中生成一个编译好的最终模板字符串然后把模板字符串渲染到内存中的DOM，此时只是在内存中渲染好了模板，并没有挂载到真正的页面上去。
2. beforeMount在beforeMount生命周期函数，表示模板已经在内存中编译完成了，但尚未把模板渲染到页面上去，页面中的元素没有被真正替换过来，只是之前的模板字符串(页面还是旧的页面)
3. 将内存中的模板真正的替换到页面上去。
4. mount在mount生命周期执时，表示内存中的模板已经真实挂载到了页面中，用户可以看到渲染好的页面了。

执行这个函数的时候就表示实例已经完整的创建完成了。

如果需要通过插件操作页面上的DOM节点，最早在mounted中进行

上面是创建阶段的函数

之后进入了运行阶段的生命周期函数

1. 执行beforeUpdate时候页面中的数据是旧的，data数据是最新的，页面中的数据和data中的数据并未同步。
2. 执行updated执行这个函数的时候数据已经和页面同步了，都是最新的。
3. BeforeDstroy钩子函数执行的时候vue实例从运行越短进入了销毁阶段，实例身上所有的data和所有的methods，过滤器等都还可以执行 还没真正的执行到销毁。
4. dstroy组件已经被完全销毁了此时所有的指令，数据，函数，过滤器都不可用了。