Vue.js

1. 概述

本文档介绍了vuejs的常用功能，让初学者快速掌握该框架的基本用法，并且运用到实际开发中。

1. Vue.js 是什么

Vue.js（读音 /vjuː/, 类似于 view） 是一套构建用户界面的渐进式框架，只关注视图层， 采用自底向上增量开发的设计，目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。

<https://cn.vuejs.org/v2/guide/> vue官方文档

<http://www.runoob.com/vue2/vue-tutorial.html> vue菜鸟教程文档（比官方文档更简洁）

<https://github.com/vuejs/vue-touch/tree/next> touch文档

1. 依赖

<!-- vue框架所需要的资源 -->

<script src="http://cdn.mingsoft.net/plugins/hammerjs/2.0.8/hammer.min.js"></script>

<script src="http://cdn.mingsoft.net/plugins/vue/2.3.3/vue.min.js"></script>

<!—基于vue的touch插件-->

<script src="http://cdn.mingsoft.net/plugins/vue.touch/2.0/vue-touch.js"></script>

1. 基本语法

Vue.js 的核心是一个允许采用简洁的模板语法来声明式的将数据渲染进 DOM，数据绑定最常见的形式就是使用 {{...}}（双大括号）的文本插值：

HTML部分：

|  |
| --- |
| <div id="app"> <!—在运用vue的区域设定一个id-->  {{ message }} <!—用 {{ 参数 }} 的形式获取内容-->  </div> |

JS部分：

var app = new Vue({

el: '#app', //对控制部分进行指定

data: { //数据声明或赋值

message: '<h1>Hello Vue!</h1>'//将页面上定义的对应 {{ message }} 参数进行渲染

}

})

页面展示效果：

Hello Vue!

1. 指令的用法 v-

* v-text 更新元素的文本内容 <div v-text=”message”></div>
* v-html 更新元素的innerHTML
* v-show 根据表达式之真假值，切换元素display css属性
* v-bind 缩写 : 动态的绑定一个或多个特性
* v-model 在表单控件或者组件上创建双向数据绑定
* v-for 基于源数据多次渲染元素活模板块

使用for语句时，用<template v-for=" "> 循环内容 </ template>作为模板，这样循环加载后，有内容才显示不会在加载过程中刷新到原有DOM内容。

* v-if 根据表达式的真假条件渲染元素，为v-if 或者v-else-if 添加else块
* v-on 缩写@，绑定事件监听器，事件类型由参数指定
* v-pre 跳过这个元素和它的子元素的编译过程
* v-cloak 这个指令保持在元素上直到关联实例编译结束
* v-once 只渲染元素和组件一次

1. 绑定属性

绑定属性：

<a v-bind:href=”indexUrl”>铭飞科技</a> 缩写形式为:href

<img v-bind:src=”picUrl” /> 缩写形式为:src

绑定样式：

<div v-bind:class="{active: isActive}" class=”active”></div>

上面的语法表示 classactive 的更新将取决于数据属性 isActive 是否为真值

1. 嵌套使用指令

* 双重循环

经常用到嵌套循环渲染数据，比如 在一个html页面上渲染 订单列表，及每个订单中的商品列表，在循环遍历订单时，需要对这个订单中全部商品也进行遍历。

HTML部分（只是主要知识点展示，不是实际订单列表中的全部代码）：

<!—用for循环取出订单列表，用item遍历取出orderList中的数据-->

<template v-for="item in orderList">

<div class="ms-order-number">订单号：

<!—补充两个小知识：获取到的地址一般是相对地址 需要补上路径，可以设置一个变量补在获取地址前-->

<a class="am-inline ms-order-number" :href="orderDetailUrl + item.orderNo">

{{item.orderNo}}

</a>

<!—再循环获取该订单的全部商品 因为商品信息在orderList中goods的子对象中，所以遍历对象是 item.goods-->

<template v-for="goods in item.goods">

<div class="am-list-news-bd">

<ul class="am-list ms-order-ul-list">

<li class="am-g am-list-item-desced am-list-item-thumbed am-list-item-thumb-left ms-order-li">

<div class="am-u-sm-4 am-list-thumb">

<a :href="baseUrl + goods.goodsUrl">

<img :src='baseHost + goods.goodsThumbnail'/>

</a>

</div>

</li>

</ul>

</div>

</template>

</template>

JS部分：

<script type="text/javascript">

ms.load(["ms","ms.mall"],function(ms,mmall) {

var orderVue = new Vue({

el: '#vue-ms-order-list',

data: {

orderList:"", //定义接返回订单列表的数组进行循环展示到页面的变量

baseHost :"{ms:global.host/}/", //定义一个基础地址，主要用于加在返回图片地址前

baseUrl :"{ms:global.url/}/m/", //定义一个基础地址，主要用于加在返回页面地址前

orderDetailUrl:"{ms:global.host/}/people/orderDetail.do?orderNo=",

},

methods:{

//获取全部订单

getOrder:function(){

var \_obj = this;

var ListParam = "orderNo=&orderStatus=-1&orderExpress=-1&orderPayment=-1&pageSize=1000&pageNo=1";

//订单列表接口

mmall.people.mall.order.list(ListParam,function(json){

if (json.length == 0 ){

alert("什么也没有")

}else{

\_obj.orderList = json.list; //将返回的json.list赋值到orderList上进行页面循环渲染

}

})

}

})

orderVue.getOrder(); //执行获取订单方法

})

</script>

页面展现效果：



* v-for循环加v-if判断

在一个循环中，可能因为数据的不同需要进行判断显示，所以需要用到循环加判断的组合。

组合语法如下：

<ul>

<li v-for=”file in files”>

<div v-if=”file.id == ‘1’”>开始</div>

<div v-else-if=”file.id == ‘2’”>暂停</div>

<div v-else=”file.id == ‘3’”>结束</div>

</li>

</ul>

1. 过滤器

* 用于过滤循环显示个数，语法如下：

Html部分：

<p v-for=”item in fileredItems”>{{item}}</p>

js部分：

conputed: {

fileredItems: function(){

return this.items.slice(0,10)//第一个参数为偏移量(第几条)，第二个参数为数量

//也可以 return this.slice(10) //默认直接显示10条

}

}

* 用于事件的延迟加载，语法如下：

html部分：

<input v-on:keyup=”doStuff”>

Js部分：

methods:{

doStuff: \_.debounce(function (){

//...

},500)//500的单位是ms

}

* 用于过滤显示的内容,语法如下：

html部分:

<p v-for=”user in filteredUsers”>{{user.name}}</p>

js部分:

computed: {

filteredUsers: function(){

var self = this;

self.users.filter(function (user){

var searchRegex = new RegExp(self.searchQuery, ’i’)

return user.isActive &&(searchRegex.test(user.name) || searchRegex.test(user.email))//通过匹配用户名和邮箱地址找到用户

})

}

}

* 用于定义一个参数让循环的数组升序或降序显示

html部分:

<p v-for=”user in orderedUsers”>{{user.name}}</p>

js部分:

computed: {

orderedUsers: function (){

Return \_.orderBy(this.users, ’name’,1)//name后面的值>=0的时候是升序<0时是降序

}

}

1. 表单控件绑定

详细案例演示参考官网：<https://cn.vuejs.org/v2/guide/forms.html#文本>

1. 基础用法

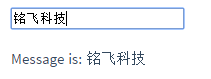
用 v-model 指令在表单控件元素上创建双向数据绑定。

html部分:

<input v-model="message" placeholder="edit me">

<p>Message is: {{ message }}</p>

页面展现效果：



1. 复选框

* 单个勾选框

Html部分：

<input type="checkbox" id="checkbox" v-model="checked">

<label for="checkbox">{{ checked }}</label>

页面展现效果：

当未勾选时绑定的checked为false，勾选后为true，下面截图为两种情况的显示

* 多个勾选框，绑定到同一个数组：

Html部分：

<input type="checkbox" id="jack" value="Jack" v-model="checkedNames">

<label for="jack">Jack</label>

<input type="checkbox" id="john" value="John" v-model="checkedNames">

<label for="john">John</label>

<input type="checkbox" id="mike" value="Mike" v-model="checkedNames">

<label for="mike">Mike</label>

<br>

<span>Checked names: {{ checkedNames }}</span>

Js部分：

new Vue({

el: '...',

data: {

checkedNames: [] //将绑定的model定义为一个空数组

}

})

页面展现效果：已勾选的选项value就会组成数组



1. 单选按钮

Html部分：

<div id="example-4">

<input type="radio" id="one" value="One" v-model="picked">

<label for="one">One</label>

<br>

<input type="radio" id="two" value="Two" v-model="picked">

<label for="two">Two</label>

<br>

<span>Picked: {{ picked }}</span>

</div>

js部分：

new Vue({

el: '#example-4',

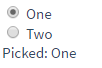
data: {

picked: ''

}

})

页面展现效果：获取到已勾选的选项value



1. 选择列表

* 单选列表

Html部分：

<div id="example-5">

<select v-model="selected">

<option disabled value="">请选择</option>

<option>A</option>

<option>B</option>

<option>C</option>

</select>

<span>Selected: {{ selected }}</span>

</div>

Js部分：

new Vue({

el: '...',

data: {

selected: ''

}

})

页面展现效果：获取到已选中的选项option的text



* 多选列表(绑定到一个数组)

Html部分：

<div id="example-6">

<select v-model="selected" multiple style="width: 50px">

<option>A</option>

<option>B</option>

<option>C</option>

</select>

<br>

<span>Selected: {{ selected }}</span>

</div>

Js部分：

new Vue({

el: '#example-6',

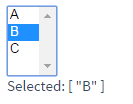
data: {

selected: []

}

})

页面展现效果：获取到已选中的选项option的text



1. 常见错误
2. Vuejs是一个严格的框架，如果没有按照规范进行使用页面将报错或者整个页面刷新为空白，下面为使用vue中常见的问题：

* 缺少逗号
* 缺少对DOM中使用的变量声明
* 单词编写错误

1. 定义了方法后，页面上没有效果

* 方法没有加载，页面加载后执行或者通过事件执行