**VUE**

# vue插件

vuex组件通信

vux组件插件

vue-resource/axios阿贾克斯请求

vue-router路由

# 构造器

new Vue({

el:’#demo’,

data:{

//数据

},

methods:{

//函数

},

directives:{

//指令

focus(){}

},

component:{

//组件

},

filters:{

//过滤器

},

store//vuex注册

router//路由注册

})

# 指令

**:**表示变量

:class=”{left:bool}” bool是变量

<!-- 指令里面不写{{}} -->

## 1变量{{}}写法

<p style="color:red">1.{{}}</p>

<p>{{name}}</p>

## 2v-text

<p style="color:red">2.v-text/ng-bind</p>

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

## 3v-html

<p style="color:red">3.v-html/ng-bind-html</p>

<!-- 安全处理 -->

<p>{{{html}}}</p>

## 4n-if

<p style="color:red">4.v-if/ng-if</p>

<p v-if="bool">Hello 假</p>

<p v-if="!bool">Hello 真</p>

## 5v-show

<p style="color:red">5.v-show/ng-show</p>

<p v-show="bool">Hello 假</p>

<p v-show="!bool">Hello 真</p>

## 6v-else

<p style="color:red">6.v-else</p>

<!--v-else和v-if/v-show要写在一起-->

<p v-if="bool">Hello 假</p>

<p v-else>Hello 真</p>

<p v-if="num>4">Hello 假</p>

<p v-else>Hello 真</p>

## 7v-for

<p style="color:red">7.v-for/v-repeat</p>

<ul>

<!--如果重复有相同的内容使用track-by="$index"-->

<li v-for="a in arr" track-by="$index" state="{{$index}}">{{a}}</li>

<li v-for="(index a) in arr" track-by="index"></li>

</ul>

## 8v-on也可以写成@click @keyup

<p style="color:red">8.v-on:click/ng-click</p>

<!-- <button v-on:click="ok()">ok</button> -->

<button @click="ok()">ok</button>

<input type="text" v-on:keyup="keyup">

## 9v-bind

<p style="color:red">9.v-bind/ng-class&&ng-style</p>

//v-bind:class -->:class

<p v-bind:class="{blue:bool}">我是class</p>

<p v-bind:style="{fontSize:size+'px'}">我是style</p>

## 10v-model

<!-- 双向数据绑定 -->

<p style="color:red">10.v-model/ng-model</p>

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

<textarea v-model="text"></textarea>

<p>你输入了:{{text}}</p>

<select v-model="text">

<option v-for="a in arr" track-by="$index" value="{{a}}">{{a}}</option>

</select>

<input type="range" v-model="size">

<p>我是obj: {{obj.name}} {{obj.age}} {{obj.skill}}</p>

<!--支持三元表达式-->

<p></p>

# 过滤器

**<!--vue v2.0只保留了自定义过滤器-->**

<p style="color:red">过滤器</p>

## currency

<p style="color:red">currency</p>

<p>{{num | currency "￥" 5}}</p>

## uppercase && lowercase

<p style="color:red">大小写</p>

<p>{{name | uppercase}}</p>

<p>{{name | lowercase}}</p>

## debounce延迟

<p style="color:red">debounce</p>

<button @click="click() | debounce 2000">debounce</button>

## orderBy && filter && limitBy

<p style="color:red">orderBy&&filterBy&&limitBy</p>

<p style="color:yellow">orderBy</p>

<ul>

//orderBy 最后一个参数是1 或者 -1

<li v-for="ele in arr1 | orderBy 'name' 1">{{ele.name}} {{ele.skill}}</li>

</ul>

<p style="color:yellow">filterBy</p>

<ul>

<li v-for="ele in arr1 | filterBy 'YANG' in 'name'">{{ele.name}} {{ele.skill}}</li>

</ul>

<p style="color:yellow">limitBy</p>

<ul>

<li v-for="ele in arr1 | limitBy 2 1">{{ele.name}} {{ele.skill}}</li>

</ul>

# 自定义指令

Vue.directive(‘name’,function(){

//钩子函数

bind(el,binding,vnode){

},

inserted(el,binding,vnode){

}

…

})

# 自定义过滤器

Vue.filter('ed',function(input){

return input + 'ed'

})

# 自定义组件

Vue.component(‘name’,{

template:’’

data:function(){

return {

//数据

}

},

props:[‘message’]//父==>子传递信息

})

# 路由

v层需要写<router-view></router-view>

var router = new VueRouter({

routes:[{

path:’/home’,

component:{

},

children:[{

//子路由没有/

path:'a',

component:{

template:'<div>a</div>'

}

}],

//重定向

{

path:’/’,

redirect:’/index’

}

}]

})

## 自定义路由对象

router.beforeEach(function(transition){

if(transition.to.path === '/index'){

//处理路由逻辑

}else{

//处理路由逻辑

}

});

路由地址跟angular一样可以忽略id=，但是需要在路由routes里面定义

例如

/home/:id/:user,在组件中可以通过this.$route.params.id获取,

/home?id=1,此情况需要通过this.$route.query.id获取

# 生命周期

beforeCreate🡺created🡺beforeMount🡺mounted🡺beforeUpdate🡺updated

# vuex

var store = new Vuex.Store({

//定义一个状态

//页面一进来就有状态,存储数据，刷新返回初始值

state:{

count:1

},

//分发状态

mutations:{

setCount:function(state,data){

state.conut = data;

}

},

actions:{

setCount(context){

context.commit(‘setCount,data)//提交触发mutations

}

}

})

new Vue({

methods:{

setCount(){

console.log(this.$store.state.count)

}

}

})

//修改state数据

// 暴力修改状态

this.$store.state.search = this.searchTitle;

// 提交修改触发mutations

this.$store.commit("setCount",this.searchTitle)

// 提交修改触发actions

this.$store.dispatch('setChange',this.searchTitle)

# vue-resource

this.Vue.http.get('http://localhost:12345/',{

}).then(function(data){

state.list = data.body;

console.log(state.list)

},function(err){})