■Vue基础

见文件vueBase1.html

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title></title>

<style type="text/css">

.cla1{

width: 100px;

height: 50px;

background-color: lawngreen;

}

.cla2{

border: solid 2px blue;

}

[v-cloak]{

display: none;

}

</style>

</head>

<body>

<!--总结开始

利用new Vue来创建Vue实例

实例中的选项:

el 选择器用来指定该实例需要挂载的元素

data 实例中定义的名值对数据都保存在这里

methods 实例中定义的方法都保存在这里

生命周期钩子，用来在特定时机触发指令如created, mounted, beforeDestroy

filters 设置对插值的过滤方法然后使用{{变量 | 过滤器名}}来实现过滤插值

computed 计算属性

指令：

v-bind的用法，用来动态更新HTML元素上的属性

v-bind绑定属性

v-bind绑定class

v-bind绑定内联样式

v-on的用法，给元素绑定事件

修饰符，这里略，详细见书p46

v-model的用法，用于双向数据绑定

v-pre的用法

v-html的用法

v-cloak的用法

v-if的用法

v-show的用法

v-if和v-show的区别

{{}}插值方法，里面的内容可以使用表达式

$el的用法

总结结束-->

<div id="app">

<input type="text" v-model="name"/> <!--v-model用来实现双向数据绑定-->

<p v-once>{{shuoming}}</p> <!--v-once表示只渲染一次，所以之后修改shuoming的值不会影响这里-->

<p v-cloak>{{shuoming}}</p> <!--v-cloak使系统加载时，不会出现未编译好的带大括号的内容。-->

<h2>你好，{{name}}</h2>

<p v-pre>这里的内容不会被编译：{{name}}</p> <!--v-pre属性表示{{}}内的内容不会被编译-->

<p>{{link}}</p>

<p v-html="link"></p> <!--v-html属性可使编译后的内容按照HTML规则输出-->

<h1>插值表达式计算后的日期：{{date.toLocaleString().slice(0,10)}}</h1> <!--使用{{}}插值，里面的内容可以使用表达式-->

<h1>过滤后的日期：{{date | jiequ}}</h1> <!--使用‘|’加过滤器名来实现过滤-->

<p>计算属性的结果：{{reversedText}}</p>

</div>

<!--使用v-bind来绑定属性及简写形式-->

<div id="app2">

<a v-bind:href="url">链接</a>

<a :href="url">简写形式</a>

<br/>

<img v-bind:src="imgUrl"/>

<img :src="imgUrl"/>

</div>

<!--使用v-bind指令绑定class-->

<div id="app3">

<!--对象语法，cla1为class名，c1为data里的变量，true是绑，false是不绑，便于之后操作-->

<div :class="{'cla1':c1, 'cla2':c2}">对象语法</div>

<!--数组语法，shu1为data里的变量，对应的值是class名-->

<div :class="[shu1, shu2]">数组语法</div>

<!--内联样式，chicun为data里的变量-->

<div :style="{'width':chicun +'px', 'height':chicun + 'px', 'background-color':colorr}">内联样式</div>

</div>

<!--v-if和v-show-->

<div id="app5">

<p v-if="status===1">使用v-if的例子1</p> <!--判断data里的变量status如果等于1时显示-->

<p v-if="panduanshow">使用v-if的例子2</p> <!--panduanshow是data里的变量，值是true时则显示，false时不显示-->

<p v-show="status===1">使用v-show的例子1</p>

<!--v-if是真正渲染，不显示就是不渲染出HTML元素。而v-show是给已有的元素加上display:none,下面的例子看控制台打印结果-->

<p id="p1" v-if="status===2">v-if和v-show的区别</p>

<p id="p2" v-show="status===2">v-if和v-show的区别</p>

</div>

<!--使用v-for遍历数组和对象-->

<div id="app6">

<ul>

<!--遍历数组-->

<li v-for="book in books">{{book.name}}</li>

<li v-for="(book, index) in books">{{index}} \* {{book.name}}</li>

<!--遍历对象-->

<li v-for="value in user">{{value}}</li>

<li v-for="(value, key, index) in user">{{index}} \* {{key}} : {{value}}</li>

</ul>

</div>

<!--使用v-on来绑定事件-->

<div id="app7">

<h1>点击次数：{{counter}}</h1>

<button v-on:click="handleAdd()">+1</button> <!--v-on简写形式是@，如下面的例子-->

<button @click="handleAdd(10)">+10</button>

<button @click="counter++">直接在@click里使用表达式的方法来实现+1</button>

<a href="http://www.apple.com" @click="handleClick('禁止打开', $event)">阻止默认行为的例子</a>

</div>

<!--表单与v-model-->

<div id="app8">

<input type="text" v-model="message" placeholder="请输入..."/>

<p>输入的内容是：{{message}}</p>

<!--单选按钮-->

<input type="radio" :checked="picked1"/>

<label>单独的单选按钮</label>

<br/>

<input type="radio" v-model="picked2" value="html" id="html"/>

<label for="html">组合的单选按钮：HTML</label>

<input type="radio" v-model="picked2" value="js" id="js"/>

<label for="js">组合的单选按钮：js</label>

<input type="radio" v-model="picked2" value="css" id="css"/>

<label for="css">组合的单选按钮：css</label>

<p>您的选择是：{{picked2}}</p>

<!--复选框-->

<input type="checkbox" v-model="checked1" id="checked1"/>

<label for="checked1">单独的复选框{{checked1}}</label>

<br />

<input type="checkbox" v-model="checked2" id="html" value="html"/>

<label for="html">组合的复选框HTML</label>

<input type="checkbox" v-model="checked2" id="js" value="js"/>

<label for="js">组合的复选框HTML</label>

<input type="checkbox" v-model="checked2" id="css" value="css"/>

<label for="css">组合的复选框HTML</label>

<p>您的选择是：{{checked2}}</p>

<!--选择列表-->

<select v-model="selected1">

<option>html</option>

<option value="js">JavaScript</option>

<option>css</option>

</select>

<p>您的选择是：{{selected1}}</p> <!--如果选中的option有value属性，则这里解析的是value的值，如果没有则解析标签内的text-->

<br />

<select v-model="selected2" multiple> <!--multiple属性可使选项变为多选-->

<option>html</option>

<option value="js">JavaScript</option>

<option>css</option>

</select>

<p>您的选择是：{{selected2}}</p> <!--如果选中的option有value属性，则这里解析的是value的值，如果没有则解析标签内的text-->

<br />

<!--动态输出option-->

<select v-model="selected3">

<option v-for="option in options" :value="option.value">{{option.text}}</option>

</select>

<p>您的选择是：{{selected3}}</p>

</div>

<script src="https://cdn.jsdelivr.net/npm/vue"></script>

<script type="text/javascript">

var ap1 = new Vue({

// el:'#app',

el: document.getElementById('app'), //el选项可以使用CSS选择器，也可以直接写el:'#app'

data:{

name:'',

shuoming:'提示内容',

link:'<a href="#">这是一个链接</a>',

date:new Date(),

text:'123, 456, 789'

},

mounted:function(){ //生命周期钩子，如created, mounted, beforeDestroy

console.log(this.shuoming); //this指向调用这个钩子的Vue实例，这里就是对象app1

console.log(this.$el.innerText); //app1.$el可以访问Vue实例挂载的元素

},

filters:{

jiequ:function(value){ //value指的是要被过滤的数据，过滤结果要用return返回出去

return value.toLocaleString().slice(0,10);

}

},

computed:{ //计算属性，结果要用return返回出去

reversedText:function(){

return this.text.split(',').reverse().join('\*');

}

}

});

ap1.shuoming="你输入的名字会同时出现在下方"; //可以修改vue实例对象中的值

var ap2 = new Vue({

el:'#app2',

data:{

url:'https://github.com',

imgUrl:'https://timgsa.baidu.com/timg?image&quality=80&size=b9999\_10000&sec=1515653478116&di=86acdac224ae6c9eda78eba853a6b422&imgtype=0&src=http%3A%2F%2Fcdn.qiaobutang.com%2Fuploads%2Fbbs\_images%2F2016%2F9%2F20%2F10%2F57e0a3120cf2da59292896b2%2Foriginal.jpg'

}

});

var ap3 = new Vue({

el:'#app3',

data:{

c1:true, //用来控制对象语法设置的class

c2:true,

shu1:'cla1', //使用数组语法设置class

shu2:'cla2',

colorr:'blue',

chicun:100

}

});

var ap5 = new Vue({

el:'#app5',

data:{

status:1,

panduanshow:true

}

});

// 以下将输出"p1元素是不存在的",因为用v-if来禁止元素显示是不渲染出元素的

if (document.getElementById("p1")) {

console.log("p1元素是存在的");

}else{

console.log("p1元素是不存在的");

};

// 以下将输出"p2元素是存在的",因为用v-show来禁止元素显示是渲染出元素然后给元素加上css display:none

if (document.getElementById("p2")) {

console.log("p2元素是存在的");

}else{

console.log("p2元素是不存在的");

};

// 另外补充,v-show不能在<template>上使用

var ap6 = new Vue({

el:'#app6',

data:{

books:[

{name:'《Vue.js实战》'},

{name:'《JavaScript语言精粹》'},

{name:'《JavaScript高级程序设计》'}

],

user:{

name:'Alice',

gender:'女',

age:26

}

}

});

ap6.books.push({name:'《新加的书》'}); //数组动态更新后，视图也会随着更新

var ap7 = new Vue({

el:'#app7',

data:{

counter:0

},

methods:{

handleAdd:function(count){

var jishu=count || 1; //如没输入参数则为1

this.counter += jishu; //this指向当前Vue实例

},

handleClick:function(message, event){

event.preventDefault();

window.alert(message);

}

}

});

var ap8 = new Vue({

el:'#app8',

data:{

message:'',

picked1:true, //单独的单选按钮，通过true和false来控制

picked2:'js',

checked1:true, //单独的复选框，通过true和false来控制

checked2:['html', 'css'],

selected1:'html', //单选的选择列表

selected2:['html', 'css'],

selected3:'html',

options:[

{

text:'HTML',

value:'html'

},

{

text:'JavaScript',

value:'js'

},

{

text:'css',

value:'css'

}

]

}

});

</script>

</body>

</html>