正课:

1. 双向绑定

2. 绑定样式

3. 计算属性

4. 自定义指令

5. 过滤器

6. axios

1. 双向绑定:

问题: :支持从Model到View的单项绑定。只能将程序中的变量值，显示到页面中。无法将页面上的更改，自动反向修改回Model的变量中。所以，对于表单元素来说，页面上元素内容修改，底层变量是无法获得的。

什么是: 即能将变量中的值，显示到页面上（M->V）。又能将页面上的任何更改自动同步到程序中的变量上（V->M）。

何时: 只要绑定表单元素的值时，都用双向绑定

如何: <表单元素 v-model:可能改变的属性="变量"

比如: <input type="text" v-model:value="kwords">

<input type="checkbox" v-model:checked="isAgree">

<select v-model:value>

原理: 其实v-model: 只不过自动给表单元素添加了oninput或onchange等事件。在事件处理函数中，自动更新Vue内的模型变量。

其实，可以实时监视一个模型变量的数值变化:

new Vue({

... ,

...,

watch:{ //监视

要监视的变量名(){

//这个函数会在模型变量的值被修改时，自动触发！

}

}

})

今后，只要希望实时根据变量的变化，自动执行操作时，都可用watch。

比如: watch:{

//只要名为kwords的变量的值发生了变化，就会自动触发监视函数。

kwords(){

//每次输入时，做的事儿和按下回车时，和点击查询按钮时，做的事儿都是一样！

this.search();

//this->凡是new Vue()中的this，都指new Vue()本身

//为什么watch中的kwords()可通过this.search()调用methods中的search()方法？

//因为不管写在哪里，最后都被升级为直接隶属于new Vue()，所以其实kwords()和search()是平级的兄弟方法！

}

}

其它表单元素的绑定:

1. 单选按钮:

<input type="radio" value="1" v-model:checked="sex">男

<input type="radio" value="0" v-model:checked="sex">女

data:{ sex:1 }

当M->V: v-model会用变量的值和当前radio的写死的value值比较。如果变量值等于当前value值，则选中。反之，则不选中

当V修改时: v-model自动将当前选中项的value值反向更新回data中的变量sex中。比如: 选择女，女的value是0，则data中sex的变量值就变成0

2. select元素

<select v-model:value="变量">

<option value="值1">文本1

<option value="值2">文本2

<option value="值3">文本3

data: { 变量: "值1" }

当M->V: v-model会用变量值和每个option的value比较。只要变量值等于某个option的value，就选中该option。

当V->M: 当select中选中项发生改变时，将当前选中的option的value，更新到程序中的变量中

3. checkbox单用

<input type="checkbox" v-model:checked="isAgree">同意

data:{ isAgree:false }

当M->V: 直接将变量中的bool值给checked属性

当V->M: 直接将变量的checked值给程序中的变量

其实: v-model可自动根据所在的表单元素的类型，判断绑定哪个属性。所以，所有v-model绑定都不需要:value或:checked。

2. 绑定样式:

1. 绑定单个css属性值: 2种:

1. 将内联样式style属性，当做一个字符串来绑定:

<ANY :style="变量">  
 data:{ 变量:"css属性:值; css属性:值;..."}

问题: 不便于修改一个css属性的值

2. 用对象方式绑定style属性:

<ANY :style="变量">

data:{

变量: {

css属性: 值,

css属性: 值

}

}

在绑定时: new Vue会自动将对象，转为style对应的字符串。

如果想修改一个css属性:

this.变量.css属性=新值

强调:

1. css属性值必须带单位

2. 如果有部分内联样式是固定不变的，则可以再写一个不带冒号的普通style，保存不变的样式。编译时，会将:style绑定的内容和不带冒号style的内容合为一个字符串共同作用域当前 元素。

何时: 只要想细致的控制某一个css属性的变化时

问题: 代码繁琐！

2. 绑定class: 2种:

1. 将class属性当做字符串绑定

问题: 如果需要绑定多个class时，字符串操作很麻烦。

2. class属性也可以用对象语法绑定:

<ANY :class="变量">

data:{

变量: {

类名1:true或false,

类名2: true或false

}

}

编译时: 只有值为true的类名，才会被编译进最终的class字符串中

如果: 变量: { 类名1: true, 类名2: true}

编译后: class="类名1 类名2"

再如果: 变量: { 类名1: false, 类名2: true}

编译后: class="类名2"

如果两个类名的值都为false！

编译后: class=""

强调: 如果有部分class是固定不变的，则可以再写一个不带冒号的普通class，保存不变的哪些class。编译时，会将:class绑定的内容和不带冒号的class的内容合为一个字符串共同作用域当前 元素。

3. 计算属性:

什么是: 自己不实际保存值，自己的属性值每次都要根据其他属性的值动态计算获得！

何时: 如果页面上需要显示一个值，但是数据中没有直接的数据用于绑定。但是却可根据其他属性值动态计算出来。

如何: 2步:

1. 在new Vue中添加计算属性:

new Vue({

el:"#app",

data:{ ... },

methods:{ ... },

watch:{ ... },

computed:{

自定义计算属性名(){

... ...

return 属性值

}

}

})

2. 在页面中像绑定普通属性一样，绑定计算属性

计算属性的用法和普通变量完全一样！

强调: 不要加()

(computed中的)计算属性 vs (methods中的)普通函数

1. 绑定时:

普通函数必须加()

计算属性不能加()

2. 当反复被绑定时:

普通函数: 绑定几次，就重复执行几次，效率低！

计算属性: 的值会被new Vue缓存起来。只要所依赖的数据不变。即使反复绑定多次，也只执行1次！效率高！

结论:

1. 如果更关心执行结果的数值时，优先用计算属性

2. 不太关心结果，更注重执行过程时，优先用普通函数

4. 自定义指令:

何时: vue自带的13种指令的功能不够用，还需要为html添加新的功能！

如何: 2步:

1. 定义自定义指令:

因为自定义指令通常可以用在所有new Vue对象中。所以，不应该在某一个new Vue里创建

应该用Vue来创建，然后所有new Vue中都能使用！

Vue.directive("指令名",{

//回调函数: 当带有这个指令的当前元素添加到DOM树上后，自动调用inserted函数

inserted(elem){

//参数elem，会自动获得当前DOM元素

//对当前DOM元素elem，执行DOM操作

}

})

强调: 指令名在定义时，一定不要加v-前缀！

2. 使用自定义指令

<ANY v-自定义指令名>

原理: 当当前元素被挂载到DOM树上时，会自动触发自定义指令中的inserted函数，并将当前元素对象<ANY>，传入inserted中。

强调: 自定义指令，只能用在受new Vue()监控的元素范围内！因为本质上，自定义指令是添加在Vue家的。只有new Vue()才认识！浏览器是不认识什么指令的！

5. 过滤器:

什么是: 对模型变量中不能直接使用的原始值进行加工后，再显示的一种函数。

为什么: 有些原始值，不能直接给用户看

比如: 时间的ms数, 性别0和1，订单状态的编号

何时: 只要原始值不能直接给用户看

如何: 2步:

1. 定义过滤器:

Vue.filter("过滤器名称",function(oldVal){  
 //加工的过程

return newVal;

})

2. 使用过滤器:

任何绑定语法中: 变量|过滤器

强调: 如果没有参数，则过滤器后不用加()

原理:

绑定语法获得变量的原始值，先传给过滤器函数的oldVal函数。

然后，过滤器函数内，对原始值进行加工，然后返回新值。

最终显示给用户的是过滤器函数返回的新值

|就像水管之间的接头一样，连同了原始的变量和过滤器函数

其实过滤器函数，也可以加参数:

定义时: Vue.filter(

"过滤器名称",

function(oldVal, 自己的形参,...){

}

)

使用过滤器时，如果需要传入参数，控制返回值

变量|过滤器(实参值,...)

强调: 实参值，不用从第二个开始。只要从第一个开始传入即可。过滤器会自动将变量的原始值作为实际的第一个实参。而我们传入的实参值，自动会从第二个开始。

多个过滤器可以串联:

变量|过滤器1|过滤器2|...

原理: 变量的原始值，先交给过滤器1处理

进过过滤器1处理后的值，不直接显示，而是继续交给过滤器2处理。

强调: 过滤器2，获得的是前一步过滤器1返回的中间处理结果。而不是原始值。

5. axios:

什么是: 专门发送http请求的基于promise技术的函数库

为什么: 不是所有情况下jQuery的$.ajax()都能用

现在想发送ajax请求:

1. ajax四步:

2. 自己封装ajax函数:

3. $.ajax(): 属于jQuery。而jQuery中不止有$.ajax()，还有非常多的DOM操作函数。——小题大做

4. Vue官方提供了vue-resource组件：不再更新。

5. Vue官方推荐: 只要Vue中发ajax请求，都用axios函数库。

何时: 今后只要在vue框架中发送ajax请求，都用axios

如何:

发送get请求:

axios.get(

"url"，

{

params:{

参数:值

}

}

).then(result=>{

//坑: axios的result不直接是服务端返回的数据。

//其实，服务端返回的数据: result.data

})