正课:

1. 指令:

2. 双向绑定:

3. 绑定class和style属性

4. 计算属性

5. 过滤器

1. 指令:

1. 绑定内容:

原始HTML片段:

问题: 不能用{{}}绑定，因为{{}}会保持HTML片段原样显示

解决: 用v-html="模型变量" 代替{{}}，可让绑定的HTML片段被浏览器解析为正文

总结: 只要绑定HTML片段，都用v-html

纯文本内容: 可使用{{}}或v-text

表单元素的值

2. v-cloak:

何时: 如果希望当new Vue加载慢时，也不让用户看到{{}}语法，就用v-cloak临时隐藏页面元素

如何: 2步:

1. 在页面的<style>中，定义属性选择器:

[v-cloak]{ display:none }

2. 在要临时隐藏的元素上添加v-cloak指令

原理: new Vue加载完之后，会自动查找v-cloak属性，并移除该属性，使临时隐藏的元素，显示出来

3. v-pre: 保持内容中的{{}}不被Vue编译

何时: 如果内容中刚好有{{}}作为正文，则需要用v-pre保护

4. v-once: 尽在首次Vue加载时，绑定一次，之后即使变量改变，也不更新页面

如何使用: <ANY v-once>绑定语法</ANY>

原理: 在首次绑定后，就将v-once所在的元素从虚拟DOM树中移除

2. 双向绑定:

问题: 普通的绑定":属性"，只能讲内存中的模型变量单向绑定到界面，无法将界面的更改，自动更新回模型变量中

解决: v-model代替:value

什么是双向绑定: 即可以将内存中的数据绑定到界面上，同时又能将界面的修改，反向更新回模型变量中

(M-V, V-M)

何时: 如果希望绑定表单元素的值时，都用双向绑定

如何: <表单元素 v-model:value="模型变量"

:value可省略

原理: Vue用一个死循环，反复扫描虚拟DOM树中的每个节点——监视

如果希望只要界面发生变化，就立刻执行操作时:

new Vue({

...,

data:{ kwords:"" },

...,

watch:{

//在kwords发生改变时，自动执行

kwords(){ }

}

})

各种表单元素的双向绑定:

1. 文本框: <input type="text"> 和文本域: <textarea>

都是绑定value

<ANY v-model="模型变量"> v-model自动绑定value属性

2. 单选按钮:

<input type="radio" name="分组名" value="值" v-model="模型变量">

分组使用:

M-V: 如果模型变量的值等于value，则当前radio选中，否则不选中

V-M: 将当前选中的radio的value值自动更新回模型变量

3. 多选框:

<input type="checkbox" v-model="模型变量"

单用: 自动绑定的是checked属性

4. select:

<select v-model="模型变量">  
 <option value="值1">

<option value="值2">

... ...

M-V: 用模型变量的值和每个option的value做比较。如果某个option的value等于模型变量的值，则该option被选中

V-M: 当选中项发生改变时，自动将选中项的value，更新到模型变量中

3. 绑定class和style:

绑定style: 2种:

1. 将style当做普通字符串属性绑定: 不直观，繁琐

2. 可用对象方式绑定style:

<ANY :style="模型变量"

new Vue({

... ,

data:{

模型变量: {

css属性:值,

... : ...

}

}

})

问题: 用style修改多个css属性，代码会很繁琐

解决:

绑定class: 2种:

1. 将class当做普通字符串属性绑定:

<ANY class="不变的class列表" :class="模型变量"

new Vue({

...,

data:{

模型变量:"可能变化的class列表"

}

})

2. 将class当做对象绑定:

<ANY class="不变的class列表" :class="模型变量"

new Vue({

...,

data:{

模型变量:{

class名: true/false,

class名:true/false

}

}

})

4. 计算属性:

什么是: 数据库中或程序中没有保存的，但需要动态计算出来的值

何时: 如果一个值，没有保存，需要每次动态计算获得

比如: 购物车的总价

如何:

new Vue({

...,

computed:{

属性名(){

...

return 属性值

}

}

})

界面中: <ANY>{{属性名}}</ANY>

其实: methods:{

方法(){ return 计算结果 }

}

界面中:<ANY>{{函数名()}}</ANY>

也行

computed vs methods:

相同: 依赖的变量发生变化，都能自动更新

不同: 1. methods要使用函数名()的调用语法

computed使用不加()的属性名即可

2. 如果需要显示多次时，

methods会重复计算多次

computed会缓存第一次计算的值，避免重复计算

5. 过滤器:

什么是: 接收原始值，经过再加工，再显示出来

何时: 只要模型变量的原始值，不能直接使用时

比如: 性别:0、1

日期/时间: ms

状态: 存数字，显示状态名

如何: