正课：

1. 指令:

2. 双向绑定:

3. 绑定class和style属性

4. 计算属性

1. 指令:

1. v-html和v-text

v-html 使用模型变量替换当前元素的innerHTML

何时: 如果绑定一段HTML代码片段

v-text 使用模型变量替换当前元素的textContent

何时: 如果绑定纯文本内容

{{}} vs v-text vs v-html

1. 是否编译内容:

v-html，专门用于绑定HTML片段，绑定的内容会被浏览器解析为正文。

{{}}和v-text，专门用于绑定不需要解析的纯文本内容，保持内容的原样不变。

2. 屏蔽{{}}

问题: 当请求响应慢时，用户可能短暂看到{{}}语法。

解决: 用v-text和v-html代替{{}}

v-text和v-html在请求响应慢时，暂时显示为空白

2. v-cloak:

什么是: 在vue实例加载完之前，暂时隐藏需要绑定的元素。

为什么: 避免在vue实例加载完之前，用户短暂看到{{}}

何时: 只要希望在vue实例加载前，不然该用户看到{{}}

如何:

2步:

1. 自定义v-cloak属性选择器，定义display:none

2. 在要绑定的元素上，添加v-cloak

原理: new Vue()实例加载完成后，会自动找到v-cloak属性，并移除。

3. v-pre: 保护内容中的{{}}不被编译，保持原样

何时: 正文中刚好也有{{}}时，不想被Vue识别

4. v-once: 让当前元素只在首次加载时，绑定一次。之后，即使模型数据发生变化，页面也不改变。

原理: 标有v-once的元素在首次加载后，就从监视队列中移除了。不再受监视。

2. 双向绑定：

单向绑定: 仅能将内存中模型数据的修改更新到视图中(M->V)

双向绑定：即能将内存中模型数据的修改更新到视图中(M->V)，又能将视图中的更改，反向更新回内存中的模型数据（V->M）

何时: 只要页面上内容修改，也想反向更新回内存中的模型数据时——专门用于绑定表单元素

如何:

<表单元素 v-model:value="模型数据"

简写: 可省略:value，<表单元素 v-model="模型数据"

自定义监听函数:

监听函数: 当模型数据发生变化时自动执行的函数

何时: 只要希望模型数据一发生变化就立刻执行一项任务——搜索帮助/表单验证

如何:

new Vue({

el:

data:

methods:

watch:{

模型变量名(){ ... } //只要同名模型变量发生变化就立刻执行

}

})

各种表单元素的绑定方式:

1. 文本框/文本域: v-model="模型变量"

2. 单选按钮:

<input type="radio" name="组名" value="值1" v-model="变量">

<input type="radio" name="组名" value="值2" v-model="变量">

其实v-model绑定的是checked属性

checked= (变量==value)

复选框: v-model绑定的也是checked属性

选择框: v-model 绑定selected= (变量==option的value)

3. 绑定class和style属性

2种方式:

1. 用字符串方式绑定class和style

2. 用对象方式绑定：

class:

1. 在new Vue({

data:{

xxxClassObj:{

fade: true/false

in: true/false

active: true/false

}

}

})

2. 在元素上: <ANY class="固定不变的class"

:class=" xxxClassObj "

结果: :class中的类会和class中的类合并

:class中只有值为true的类名才能出现在class中

总结:

1. 在多个元素之间切换一个class，首选用字符串方式

2. 控制一个元素的class，有与没有切换，首选用对象方式

style:

2种方式:

1. 作为字符串属性绑定:

2. 用对象方式:

2步:

1. new Vue({

el:

data:{

styleObj:{

css属性: 属性值,

... : ...

}

}

})

2. <ANY :style="styleObj">

建议: 今后所有style都采用对象语法绑定。

4. 计算属性：

什么是: 不实际存储属性值，而是根据其他属性的值动态计算获得。

为什么: 不是所有值都可直接获得。

何时: 只要一个值不能直接获得，需要根据其它值计算获得时

如何:

1. new Vue({

el:

data:

methods:

watch:

computed:{

属性名(){ return 计算后的值 }

}

})

2. 绑定：和普通属性的绑定完全一样！

{{属性名}}

强调: 不要加()

计算属性 vs 函数

计算属性的值可被Vue缓存并重复使用，只要计算属性依赖的其它属性不改变，就不用重复计算！

函数只要调用，就重新计算！