正课:

1. 什么是Vue

2. 如何使用Vue

3. 绑定语法

4. 指令

1. 什么是Vue

Vue是基于MVVM模式的渐进式的纯前端js框架

原生DOM vs jQuery函数库 vs 框架

原生DOM: 步骤繁琐，API长

jQuery函数库: API变得极简化，但是步骤没有减少

框架: 已经是一个半成品的项目，只不过需要加入个性的内容而已。避免了大量重复的代码。

何时: 以数据操作为主的项目

2. 如何使用Vue:

2种:

1. 下载并引入vue.js文件

官网: https://cn.vuejs.org/

版本: 2.6

开发版: 拥有完备的注释，整齐的代码结构，以及完整的错误提示信息。

生产版: 去掉所有注释，压缩代码格式为最简化，并删除所有错误提示。

如何使用:

1. 在网页中引入vue.js

2. 编写网页的HTML

强调: 网页的内容必须包含在一个唯一的父元素内

2件事:

1. 将要显示的数据变量，提前放置在页面的对应位置

2. 顺便绑定事件

3. 编写自定义脚本:

2件事：

1. 创建对象data: 定义页面上需要的所有数据变量

2. 创建Vue对象，将data和网页绑定起来

2. 脚手架代码

Vue的原理:

MVVM: 设计模式

传统的前端三大文件的划分:

HTML: 静态内容，不支持动态变化

CSS: 静态样式，不支持动态变化

JS: 所有动态变化都需要js操作

现代框架划分:

1. 界面: View视图

HTML+CSS

HTML中提前放置数据变量和事件处理函数

2. 模型数据: Model

data: 专门保存页面所需的所有数据的普通对象

3. 控制器: ViewModel

将View界面和模型数据Model绑定起来

结果: Model中的模型变量值更新，View中的元素内容自动更新

3. 绑定语法:

<ANY>{{模型变量}}</ANY>

将内存中(data中)一个模型变量的值，自动放在指定元素的内容中。

学名: 插值语法: interpolate

俗称: 大胡子语法

何时: 只要动态绑定元素的内容时

{{各种有返回值的js表达式}}

包括: 算术计算，方法调用，访问数组元素，访问对象的属性，三目运算

比如:

<h1>用户名:{{uname}}</h1>

<h2>积分:{{score}}</h2>

<h3>性别:{{sex==1?"男":"女"}}</h3>

<h3>下单时间:

{{new Date(orderTime).toLocaleString()}}</h3>

<h3>总价: ¥{{(price\*count).toFixed(2)}}</h3>

<h3>今天，星期{{week[3]}}</h3>

不能放: ~~程序结构~~: if else if else while for

4. 指令:directive

什么是: 为HTML语言添加新功能的一组自定义属性

为什么: HTML语言是静态语言，缺少程序必要的功能: 变量，判断，循环

何时: 如果希望HTML中也能使用变量，判断，循环等程序功能时，就用指令。

包括: 13种

1. 绑定属性值:

错误的做法: <img src="{{src}}"> 报错

data中变量src

正确: <img v-bind:src="src">

data中的变量src

v-bind:属性名="变量"

动态绑定指定的属性名为内存中指定的变量值

比如: <img v-bind:src="src">

将img的src属性动态绑定到data中的src变量

运行时，自动将data中src的变量值插入img的src属性中。

可简写为: ~~v-bind~~:属性名="变量"

2. 控制元素的显示和隐藏:

<ANY v-if="条件表达式">

如果满足"条件表达式"的要求，<ANY>才显示，否则就隐藏

比如: <div v-if="islogin==false">

data: { islogin: false } //此时，因为条件满足，所以div是显示的。

true//改为true，因为条件不满足，所以div不显示。

<ANY v-show="条件表达式">

鄙视: v-if vs v-show

1. v-show: 用display:none控制隐藏 —— 效率略高

v-if: 用空注释节点替换要隐藏的节点对象

2. v-if可与v-else-if和v-else组成多选一的效果

v-show只能单独使用

<ANY v-if="条件1"

<ANY v-else-if="条件2"

...

<ANY v-else>

同程序中的if ... else if... else 结构

依次判断每个元素上的条件。哪个条件满足，就显示哪个元素。其余元素不显示

强调: v-if与v-else-if与v-else之间不能插入其他元素

3. 根据数组中的元素反复生成多个相同结构的DOM元素

比如: pics:[{图片1}, {图片2},{图片3},... ...]

<ul>

<li><img src="图片1">  
 <li><img src="图片2">

<li><img src="图片3">

... ...

specs:[{规格1},{规格2},{规格3},{规格4},]

<a href="details.html?lid=1">规格1</a>

<a href="details.html?lid=2">规格2</a>

<a href="details.html?lid=3">规格3</a>

<a href="details.html?lid=4">规格4</a>

如何: <ANY v-for=" (elem,i) of 数组"

遍历data中的一个数组

每遍历数组中的一个元素

就自动创建一个<ANY>元素加入页面中

同时获得的元素值和下标，可用作绑定新元素的内容和属性

强调: v-for不能写在父元素上，而应该写在要反复生成的元素上

简写: 如果只关心元素的内容，不关心下标

v-for="elem of tasks"

4. 事件绑定:

<ANY v-on:事件名="处理函数名" //不建议加()

简写: v-on:click 可简写为@click

定义处理函数:

所有处理函数以及将来的自定义方法，都必须写在：

new Vue({

el:"#app",

data:{ 模型变量 },

methods:{

处理函数:function(){

this->当前vue对象本身

//访问vue中data中的模型变量: this.变量

不管写在data中，还是写在methods中，最后都直接属于new Vue对象。

}

}

})

5. 绑定HTML片段:

问题: {{}}不能绑定html片段

如果绑定，会原样显示，不解析

解决: v-html="变量" 代替{{}}

6. 避免网页加载慢时，用户短暂看到{{}}语法

2种:

1. v-cloak: 2步:

1. 在要隐藏的元素或父元素上，定义v-cloak属性，不给属性值。

2. 在网页的css中，添加属性选择器:

[v-cloak]{ display:none }

结果: 当vue加载完，会自动查找v-cloak属性，并移除该属性

2. <ANY v-text="`拼字符串内容`"

7. 只在页面加载时，绑定一次

之后，即使模型变量修改，页面也不再自动更新

<ANY v-once>

8. 保护内容中的{{}}不被编译

何时: 如果内容中恰好有不希望被编译的，要原样显示的{{}}，就用v-pre

总结：

1. 只要动态绑定元素的内容: {{变量/表达式}}

2. 只要动态绑定元素的属性值: :属性名="变量/表达式"

3. 控制一个元素的显示/隐藏: v-show="判断条件"

4. 控制多个元素选其一显示:

v-if="条件1"

v-else-if="条件2"

...

v-else

5. 根据数组反复生成多个相同结构的元素:

v-for="(elem,i) of 数组"

6. 只要绑定事件，就用: @事件名="处理函数名"

7. 如果绑定的内容是HTML片段，必须用v-html="变量"

8. 避免短暂看到{{}}: 2种:

<ANY v-cloak>{{变量}}

<ANY v-text="`拼内容字符串`"

常见错误:

Error compiling template: 模板编译错误

错误 编译 模板

说明HTML模板中有语法错误