小程序: WEB问题速查

1. 问题清单: 历届师兄师姐在今天课程中集中问过的高频的重点的问题！

那些自己发现不了问题，又迫切想知道其它同学都问了什么问题，的同学

小程序->首页->VUE->day01

当老师在回答其它同学的问题时，你不要闲着！你可以打开问题清单浏览别人曾经问过的经典问题！

2. 高清无码小视频: 小程序->在线->VUE->day01

2部分: 1. 课上重点知识和案例的回放！2. 重点高频笔试面试题讲解扩展视频

万一家里或中心网络断了！暂时看不了直播了！

因为某些事儿请假了，错过了当天的直播！

课下复习时，觉得TMOOC视频太大，浏览不方便或TMOOC不能看了！

——打开小程序中"在线"，就可下载观看当前所讲内容和当天所讲全部内容的回放。

基础比较好的，确实学有余力的同学，想拿高薪的同学，在我答疑时，不要闲着:

要看今天扩展视频:

小程序->在线->VUE->day01

高频笔试题: 观察者模式

3. 单词列表: 小程序->单词->VUE->day01

第三阶段上课约定:

1. 上课千万不要跟着记笔记和写代码！

2. 凡是拼写错误，自己懒得找！我可以帮你找，但是找到后，错误的单词你要手抄20遍，并拍照！

出错: 先开f12，console，看错误提示！

3. 每个带注释的案例至少写三遍:

(1). 注释+代码抄一遍

(2). 保留注释，删除代码，自己试着把代码翻译回来！

(3). 删除注释和代码，自己试着用自己的话把注释写回来！

一个例子只有自己可以把人话的注释写回来！才算自己会！

问题: 一听就会一做就废！原因: 你自己无法用人话精确描述自己想干什么！

如何提问: （vip待遇的剩余时间5天）

1. 上课时，案例中，笔记中那句话哪行代码看不懂，立刻问！

2. 小程序中问题清单里哪个问题和回答看不懂！立刻问！

3. 个人项目中哪个功能不会做！立刻问！

4. 刷笔试题时，遇到不会的笔试题！立刻问！

谁问的问题越多！谁的培训费交的越值！

正课

1. 什么是VUE

2. 如何使用VUE

3. MVVM设计模式

一. 什么是VUE

1. 什么是: 第三方开发的基于MVVM设计模式的渐进式的纯前端js框架

(1). 第三方: 下载

(2). 基于MVVM: 稍后重点讲

(3). 渐进式: 非常容易和其它技术融合，可逐步在项目中使用vue，而不要求必须整体替换！

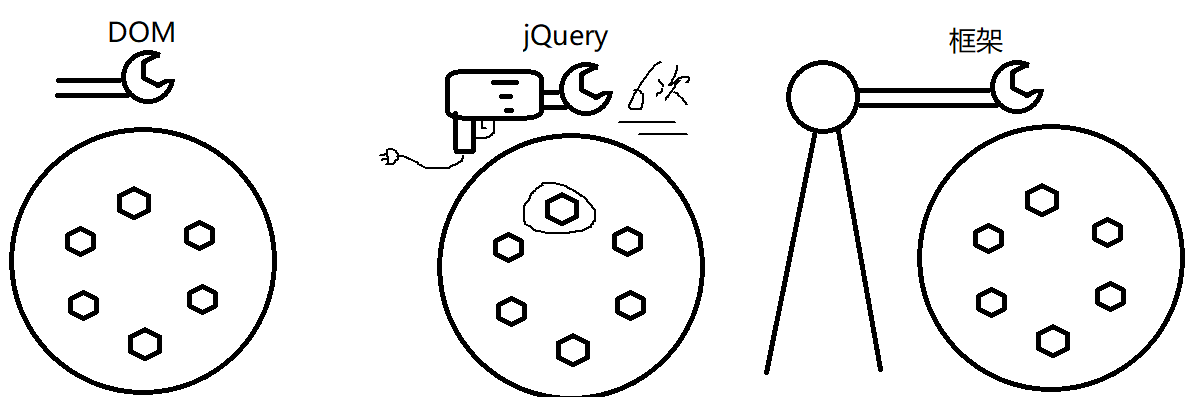
(4). 纯前端js: 不需要任何后端技术，就可独立开发和运行

(5). 框架:

a. 什么是: 已经包含部分核心功能的半成品代码，缺少个性化定制。

b. 为什么: 避免重复编码

c. 原生DOM vs jQuery vs Vue



2. 何时: 今后绝大多数**以数据操作（增删改查）为主**的PC端或移动端项目都可用vue开发

比如: 美团，饿了么，淘宝，知乎，大众点评，微博...

二. 如何使用VUE

1. 下载: 2种:

(1). 简单: 去官网下载vue.js文件，引入网页中——适合于初学者入门(3天)

cn.vuejs.org

(2). 复杂: 通过脚手架代码使用——综合，标准化，适合公司开发之用(2天)

2. 第一个vue程序: 3步:

(0). 也需要先引入vue.js

<script src="js/vue.js">

(1). 先定义增强版界面: (HTML+CSS)

a. 要求**1: 整个界面要用一个唯一的父元素包裹起来,习惯上都用<div id="app">**

b. 要求2: 找到界面中将来可能发生变化的位置，**用{{自定义变量名}}**特殊语法标记出来

c. 要求3: 找到界面中可以点击的位置绑定事件处理函数: **@事件名="自定义处理函数名"**

(2). 创建一个Vue类型的对象来监视页面中的内容

(3). 在Vue中定义好页面所需的一切资源: 变量+事件处理函数

a. 页面所需的所有**变量**，都必须保存在**data属性中**，且以**对象属性形式**保存

b. 页面所需的所有**事件处理函数**，都必须保存在**methods**属性中，且以**对象方法形式**保存

3. 示例: 使用vue实现点击+/-按钮，修改数量:

1\_first\_jq.html

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>    <meta charset="UTF-8">    <meta name="viewport" content="width=device-width, initial-scale=1.0">    <title>Document</title>    <script src="js/jquery-1.11.3.js"></script>  </head>  <body>    <button>-</button><span>0</span><button>+</button>    <script>      //点+ 数量+1      //DOM 4步      //1. 查找触发事件的元素      //查找内容中包含+的button元素      $("button:contains(+)")      //2. 绑定事件处理函数      .click(function(){        //3. 查找要修改的元素        //本例中: 查找当前按钮的前一个元素span        var $span=$(this).prev();        //4. 修改元素        //4.1 取出元素的旧内容，并转为整数        var n=parseInt($span.html());        //4.2 计算新值: 将n+1        n++;        //4.3 将新值再放回去        $span.html(n);      });      //点- 数量-1      //DOM 4步      //1. 查找触发事件的元素      //查找内容中包含-的button元素      $("button:contains(-)")      //2. 绑定事件处理函数      .click(function(){        //3. 查找要修改的元素        //本例中: 查找当前按钮的后一个元素span        var $span=$(this).next();        //4. 修改元素        //4.1 取出元素的旧内容，并转为整数        var n=parseInt($span.html());        //4.2 计算新值:        //如果n>0，才能-1        if(n>0){          n--;        }        //否则什么也不干        //4.3 将新值再放回去        $span.html(n);      });    </script>  </body>  </html>  运行结果: |

2\_first\_vue.html

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>    <meta charset="UTF-8">    <meta name="viewport" content="width=device-width, initial-scale=1.0">    <title>Document</title>    <script src="js/vue.js"></script>  </head>  <body>    <!--1. 先定义增强版界面（HTML+CSS）      要求1: 整个界面要用一个唯一的父元素包裹起来,习惯上都用<div id="app">      要求2: **找到界面中将来可能发生变化的位置，用{{变量名}}特殊语法标记出来**      本例中: 点+/-号，span元素的内容会发生变化      要求3: 找到界面中可以点击的位置绑定事件处理函数      //本例中: 点button +，执行+1(add)操作, 点button -, 执行-1(minus)操作    -->    <div id="app">      <button @click="minus">-</button>      <span>{{n}}</span>      <button @click="add">+</button>    </div>    <script>      //2. 创建一个Vue类型的对象来监视页面中的内容      //问题: 监视哪儿？      //解决: **用el属性:"选择器"，为new Vue指明要监视的范围。**el是element的缩写，指要监视的元素      new Vue({        el:"#app", //new Vue只负责id为app的div内的元素        //3. 在Vue中定义好页面所需的一切资源: 变量+事件处理函数        //3.1 页面所需的所有变量，都必须保存在data属性中，且以对象属性形式保存        data:{          //因为界面上需要一个变量n，所以这里需要些一个n支持页面          n:0 //起始值为0        },        //3.2 页面所需的所有事件处理函数，都必须保存在methods属性中，且以对象方法形式保存        methods:{          //因为界面上需要两个事件处理函数add和minus          //所以这里需要定义两个方法          add(){//add:function(){            //vue中的事件处理函数不需要做任何DOM操作，只需要专心操作data中的变量即可            //强调: **vue中methods中的方法要使用data中的变量，必须加this.**            this.n++            //结果: new Vue会自动将n变量的新值放到页面上          },          minus(){//minus:function(){            //如果当前n>0，才n--            if(this.n>0){ this.n-- }          }        }      })    </script>  </body>  </html>  运行结果: |

三. MVVM设计模式:

1. 什么是: 对前端代码的重新划分

2. 为什么:

(1). 旧的前端代码分为三部分:

a. HTML: 仅包含网页内容的静态代码

b. CSS: 仅包含网页样式的静态代码

c. JS: 负责一切网页内容、属性和样式的变化

(2). 问题：

a. HTML和CSS缺少程序必要的要素: 没有变量，没有分支，没有循环，HTML和CSS几乎生活不能自理！任何一点变化，都要靠js来完成

b. js要负责HTML和CSS中几乎所有变化，导致大量重复的代码和繁琐的步骤！

3. MVVM如何重新划分前端代码: 依然3部分:

(1). **界面**(View): 增强版的HTML+CSS

a. HTML中可以使用变量了

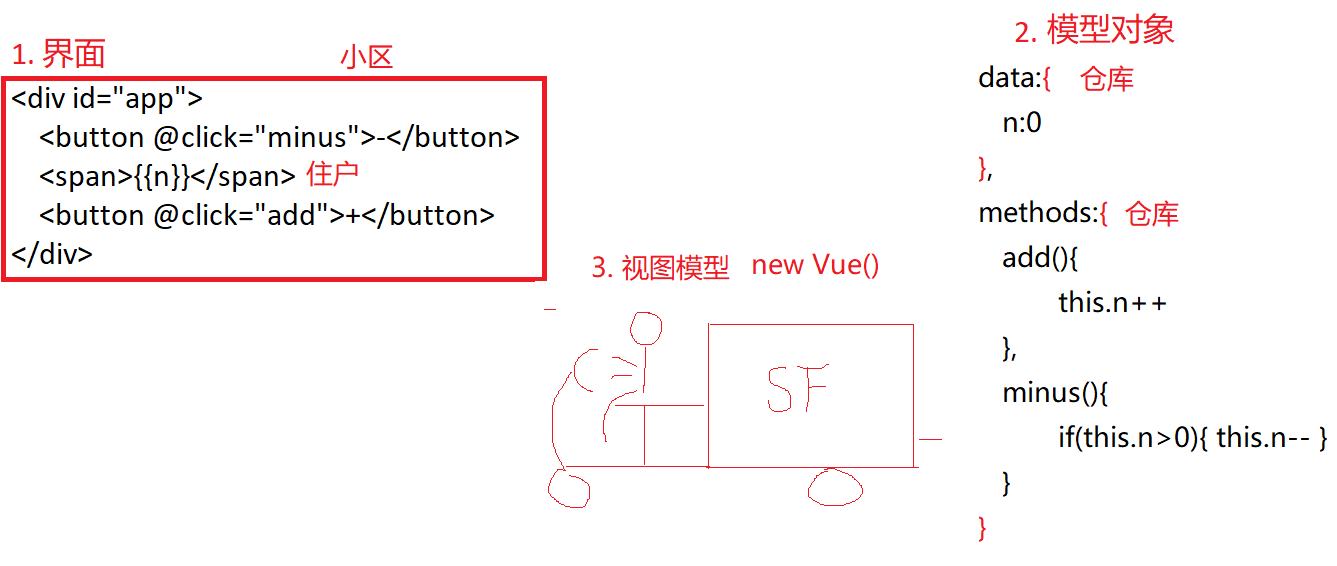
b. HTML中可以使用分支和循环了

(2). **模型对象**(Model): 专门保存界面所需的所有变量和事件处理函数的对象

a. 专门保存界面所需的一切变量的对象，称为data

b. 专门保存界面所属的一切事件处理函数的对象，称为methods

(3). **视图模型**(ViewModel): 一种专门自动将模型对象中的数据和函数与界面中保持同步 的特殊的程序



4. Vue框架如何实现MVVM设计模式的: Vue的绑定原理

(1). 引入vue.js时，其实是引入了一种类型Vue

(2). 如果向使用vue框架做开发，必须都要创建new Vue()对象

(3). new Vue()刚创建时，先把模型对象的data和methods中的数据引入到new Vue()对象中。

a. 先将data中的原变量隐姓埋名，半隐藏，再自动给data中每个变量请访问器属性(保镖)冒名顶替。结果: 今后，在任何位置只要使用变量，其实使用的都是访问器属性，已经不可能直接使用原变量了！

b. 在每个变量的访问器属性中的set函数中，都内置了一个通知函数。

c. 结果: 将来只要试图修改访问器属性表示的变量时，都会触发set中的通知，来通知整个框架，哪个变量被修改了！

d. 然后，new Vue()会将methods中的事件处理函数，全部打散。导致methods中的所有事件处理函数，变成直接隶属于new Vue()对面的方法！

e. 结果:

1). methods中的函数要想使用new Vue()中访问器属性表示的变量，必须加this.

2). methods中的函数中的this，默认都指向当前new Vue()对象

(4). 当new Vue()的data，methods和访问器属性都创建完毕后，new Vue()开始扫描el属性所指向的页面区域

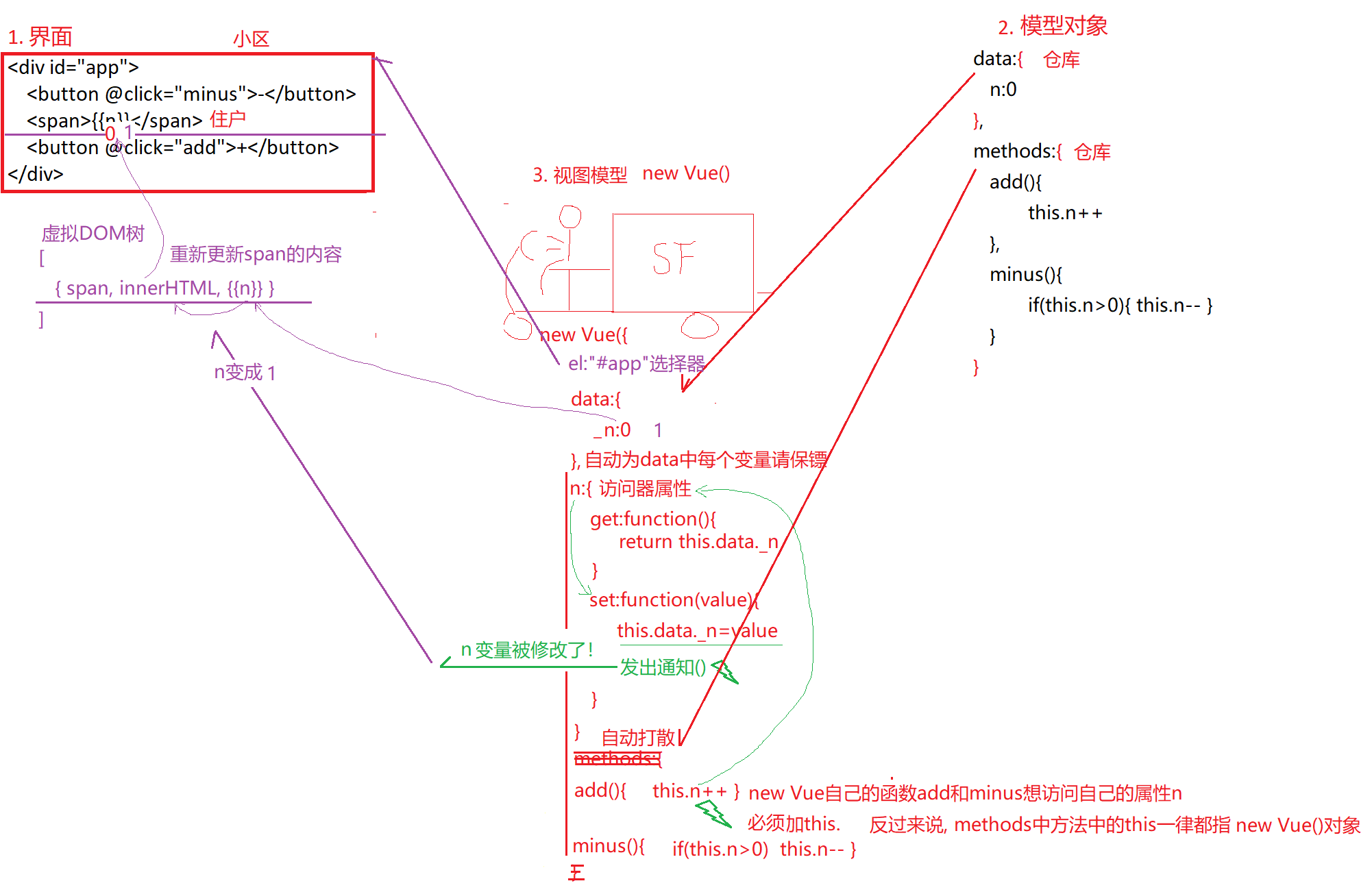
a. 扫描过程中，只会将可能发生变化的元素保存在一个集合中。

这个仅保存可能变化的少量的元素的集合，称为虚拟DOM树

b. 首次扫描时，new Vue()会将data中的变量值首次自动更新到对应的元素上

c. 今后new Vue()中任何一个变量一旦别修改，都会自动触发set函数中的通知函数，通知函数就会向虚拟DOM树发送通知.

d. 一旦虚拟DOM树接到某个变量发生变化的通知之后，就会重新扫描"虚拟DOM树"，快速找到受本次变量变化影响的元素。并用变量的新值，自动更新元素的内容。



5. 总结: **vue的绑定原理: 访问器属性+虚拟DOM树**

6. **虚拟DOM树的优点:**

(1). **小**: 只包含可能变化的元素。

(2). **遍历查找快**

(3). **修改效率高:** 只修改受影响的元素。不受影响的元素，一点儿也不改！

(4). **避免重复编码**: 已经封装了增删改查操作！程序员无需手动编写增删改查代码。

7. 总结: vue功能的开发步骤: 3步

(1). 先创建增强版的界面

(2). 再创建new Vue()对象,其中el:指向new Vue()要监控的区域

(3). 在new Vue()对象内定义模型对象data和methods

a. 所有界面所需的变量都放在data中

b. 所有界面所需的事件处理函数都放在methods中

总结:

1. MVVM: 界面View+模型Model+视图模型ViewModel

2. Vue绑定原理: 访问器属性+虚拟DOM树

变量被修改时: 访问器属性发出通知，虚拟DOM树扫描并仅更新受影响的元素

3. 虚拟DOM树优点:

(1). 小: 只包含可能变化的元素。

(2). 遍历查找快

(3). 修改效率高: 只修改受影响的元素。

(4). 避免重复编码: 已封装DOM增删改查代码

4. Vue功能3步:

(1). 先创建增强版的界面:   
a. 可能变化的元素内容用{{自定义变量名}}标记

b. 触发事件的元素用@click="自定义处理函数名"标记

(2). 再创建new Vue()对象,其中el:指向new Vue()要监控的页面区域

(3). 在new Vue()对象内定义模型对象data和methods

a.界面所需的所有变量都放在data中

b.界面所需的所有事件处理函数都放在methods中

作业:

1. 复习今日小程序问题清单: 小程序->首页->VUE->day01

2. 预习: 小程序->在线->VUE->day01

3. 绑定语法{{}} 指令 v-bind ... v-show ...v-if...v-else-if...f-else

4. v-for 遍历数组

3. (学有余力) （凡是觉得知识点串不起来）看小程序视频 完成 jQuery版学子商城项目

小程序->在线->JQUERY->day04 项目: 0~4 学子商城项目...

所有人下载项目:0. 准备中的public\_start文件夹，是项目的起始状态（空的），看着视频中项目:1，项目:2, 项目:3,...跟着做。

public.zip中是完成的代码，如果作业过程中出错，可以跟public.zip中代码对一下。

家里环境xampp和nodejs能用，且第一阶段学的好的同学:

从小程序中项目: 0. 准备...中下载xzserver.zip，用自己的服务器端代码

家里环境xampp和nodejs不能用，或者第一阶段学的不好的同学：

无需下载xzserver.zip

视频中所有ajax请求的服务器端接口地址，都改为我的新浪云服务器地址，就不需要你再运行服务器端了。也不需要你了解任何服务器端知识了:

/index 改为 http://xzserver.applinzi.com/index

/details 改为 http://xzserver.applinzi.com/details?lid=xxx