



学期项目二

（CMS）

项目报告

题 目： 乐购商城

系 别： 信息工程系

专业班级： 软件19332班

姓 名： 张成伟

指导教师： 杜灵

2021 年 1 月 1 日

目录

[一.项目总结 1](#_Toc16589)

[1.项目介绍： 1](#_Toc181)

[2.项目周期：3~6个月 1](#_Toc25208)

[3.人员配置： 1](#_Toc1900)

[4.项目完整流程 1](#_Toc28106)

[5.项目技术栈：html/h5 css/css3 javascript jquery 1](#_Toc24367)

[6、项目职责： 1](#_Toc28916)

[二.知识点总结 3](#_Toc20342)

[三.面试会遇到的问题及解决办法 5](#_Toc26483)

[1.JS数据类型有哪些？（基本类型和引用类型） 5](#_Toc30755)

[2.如何转化类型？ 5](#_Toc8846)

[3.数组从小到大排序 5](#_Toc6795)

[4.数组从大到小排序 5](#_Toc26314)

[5.显示转换与隐式转换 6](#_Toc22424)

[6.mouseover/mouseout与mouseenter/mouseleave的区别与联系 6](#_Toc28156)

[7.foreach与map和filter的区别 7](#_Toc29720)

[8.offsetWidth/offsetHeight、clienWidth/clienHeight与scrollWidth/scrollHeight的区别？ 8](#_Toc15068)

[9.==与===的区别？ 8](#_Toc31867)

[10.Null和undefined的区别？ 8](#_Toc21065)

[11.字符串操作方法（8个） 9](#_Toc11150)

[12.数组操作方法（10个） 11](#_Toc20385)

[13.JS中定时器有哪些？它们的区别及用法是什么？ 14](#_Toc21489)

[14.Document.write和innerHTML的区别？ 14](#_Toc13525)

[15.解释一下DOM和BOM？ 15](#_Toc19035)

[16.jQuery一个对象可以同时绑定多个事件，这是如何实现的？ 16](#_Toc7204)

[17.针对jQuery的优化方法？ 17](#_Toc32492)

[18.jQuery常用选择器 17](#_Toc24734)

[四.商城开发过程中遇到的问题及解决方案？ 21](#_Toc9754)

[1.HTML问题 21](#_Toc15331)

[2.Css 21](#_Toc5629)

[3.注意 21](#_Toc2749)

[4.JS如何解决问题 23](#_Toc25350)

**一.项目总结**

**1.项目介绍：**

一个商场类型的项目，主要是一个平台，用户分为商家、卖家，商家可以在这个平台发布产品，出售商品，买家可以搜索、买商品、支付、评论。项目包含了：登录注册模块、首页展示模块、商品展示模块、购物车模块、产品详情模块、会员中心模块……

**2.项目周期：**1~2个月

**3.人员配置：**

1. 项目经理：1
2. 产品经理：1
3. UI设计：1
4. 前端：1
5. 后端：4
6. 测试：1人（有些公司没有，有些公司测试是独立的）

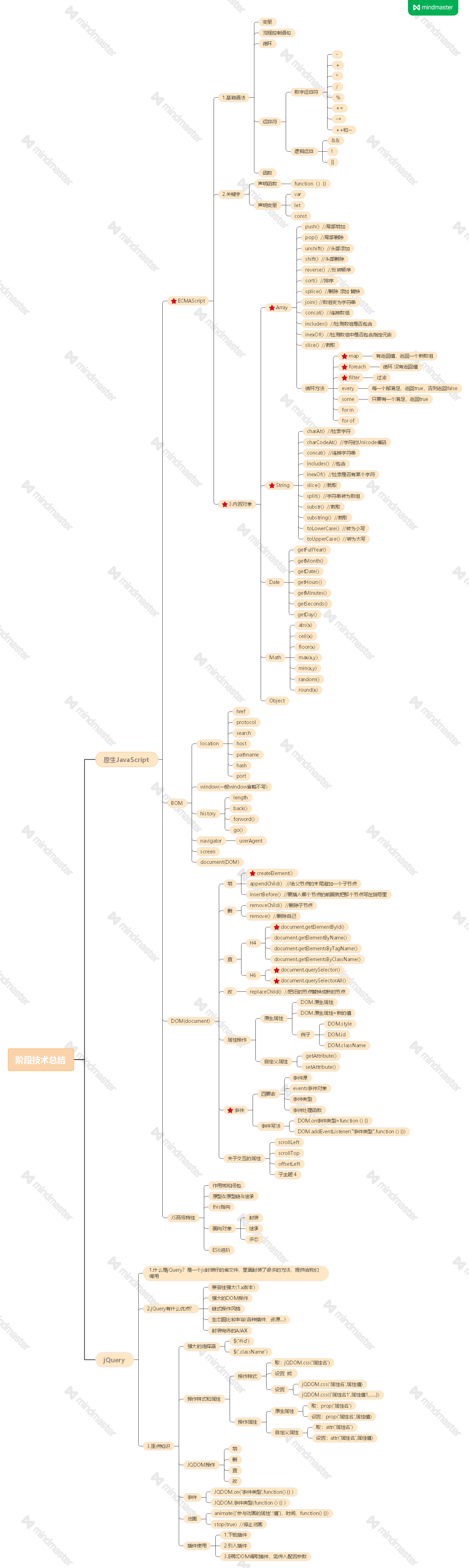
**4.项目完整流程**

需求收集=>需求评审=>项目立项=>项目排期（甘特图）=>产品原图=>UI（前端后端）=>项目测试=>项目部署上线=>后期维护（版本迭代）

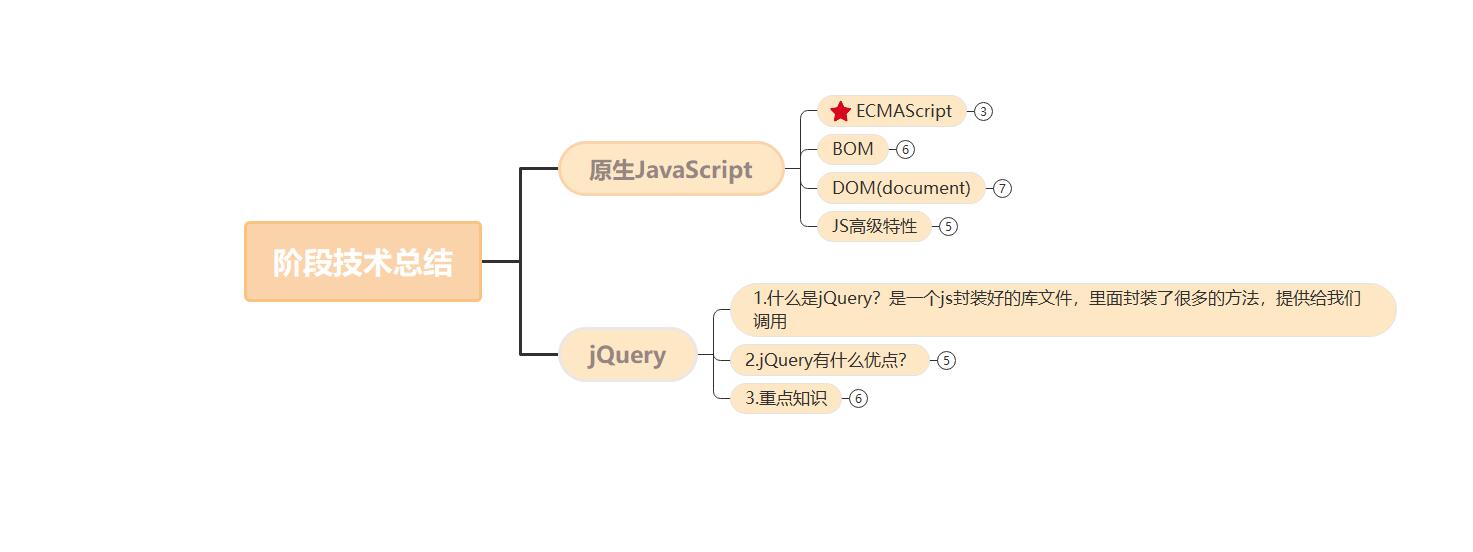
**5.项目技术栈：**Html/H5 Css/Css3 , Javascript , Jquery。

**6、项目职责：**

1. 负责了项目工程目录搭建
2. 高度还原了UI设计
3. 首页模块以及所有前端交互
4. 购物车模块以及前端交互
5. 产品详情模块以及前端交互

**二.知识点总结**

图

******三.面试会遇到的问题和解决办法**

**1.JS数据类型有哪些？（基本类型和引用类型）**

**1、栈(stack)和堆（heap）**

stack为自动分配的内存空间，它由系统自动释放；而heap则是动态分配的内存，大小也不一定会自动释放

**2、数据类型**

　　JS分两种数据类型：

**基本数据类型**：**Number、String、Boolean、Null、 Undefined、Symbol（ES6），**这些类型可以直接操作保存在变量中的实际值。

**引用数据类型**：**Object（在JS中除了基本数据类型以外的都是对象，数据是对象，函数是对象，正则表达式是对象）**

**3、基本数据类型（存放在栈中）**

     基本数据类型是指存放在**栈**中的**简单数据段，数据大小确定，内存空间大小可以分配，**它们是直接按值存放的，所以可以直接**按值访问**

**4、引用数据类型（存放在堆内存中的对象，每个空间大小不一样，要根据情况进行特定的配置）**

  引用类型是存放在堆内存中的对象，变量其实是保存的在栈内存中的一个指针（保存的是堆内存中的引用地址），这个指针指向堆内存。

    引用类型数据在栈内存中保存的实际上是对象在堆内存中的引用地址。通过这个引用地址可以快速查找到保存中堆内存中的对象

**5、浅拷贝**

前面已经提到，在定义一个对象或数组时，变量存放的往往只是一个地址。当我们使用对象拷贝时，如果**属性是对象或数组时**，这时候我们传递的也只是一个地址。因此子对象在访问该属性时，会根据地址回溯到父对象指向的堆内存中，即**父子对象发生了关联**，两者的属性值会指向同一内存空间。

**6、深拷贝**

或许以上并不是我们在实际编码中想要的结果，我们**不希望父子对象之间产生关联**，那么这时候可以用到**深拷贝**。既然属性值类型是数组和或象时只会传址，那么我们就用**递归**来解决这个问题，把父对象中所有属于对象的属性类型都遍历赋给子对象即可。

**2.如何转化类型？**

Js中的数据类型一共有六种，即number，string，boolean，underfine，null，object。

一，number

Number数据类型指的是数字，可以为整型，也可以是浮点数。

二，string

字符串由零个或多个字符构成，字符包括字母，数字，标点符号和空格;需要注意的是

字符串必须放在引号里（单引号或双引号）;

三，boolean

布尔型数据只能有两种值 true 和 false，在js中true和false是关键字。通常，我们设置布尔值时多用于条件的判断。

四，underfine

underfine通常指的是没有赋值的变量，通过typeof可以对数据的类型进行判断。

五，null

null是一个只有一个值的特殊类型。表示一个空对象引用。如：

var a=null;

将a清空。

六，object

对象就是由一些彼此相关的属性和方法集合在一起而构成的一个数据实体。常见的对象有array，window，document等。

**3.数组从小到大排序**

按照数字大小排序，小的在前面，大的在后面。

**4.数组从大到小排序**

按照数字大小排序，大的在前面，小的在后面。

**5.显示转换与隐式转换**

显式是强制转换bai，而隐du式是编译器自动转换的zhi，一般同一dao数据类型（如byte,int,float,double），由低zhuan级转为shu高级为隐式（如int转为float或double），而高级转为低级为强制类型转换，即显式转换，须用户在代码中写的（如float转为int）。

隐式数据类型转换是有编译器完成的，例如：

char a = 'a';

int x = a;

显示需要编程过程中强制转换，例如

int a = 10;

double x = (double)a;

**6.mouseover/mouseout与mouseenter/mouseleave的区别与联系**

1).mouseover:鼠标指针穿过被选择元素或其子元素，均会触发事件；

2).mouseenter：鼠标指针穿过被选择元素时才触发事件；

3).mouseout:鼠标指针离开被选择元素或其子元素则触发事件；

4).mouseleave：鼠标指针离开被选择元素时才触发事件

mouseover/mouseout是标准事件，支持所有的浏览器；mouseenter/mouseleave是IE5.5引入的特有事件，后来被DOM3标准采纳，现代浏览器也支持。

mouseover/mouseout是冒泡事件；mouseenter/mouseleave不冒泡。需要为多个元素监听鼠标移入/移出事件时，推荐使用mouseover、mouseout托管，提高性能。

若是单个元素，鼠标指针移入移出，mouseover/mouseout和mouseenter/mouseleave事件的触发效果一致；若是嵌套元素，mouseover/mouseout会往子级传递，并且鼠标指针从父级进入子级或者从子级进入父级时，都会触发离开的mouseout事件和进入的mouseover事件，而mouseenter/mouseleave只会在绑定的元素上生效。

**7.foreach与map和filter的区别**

foreach：没有返回值，只是对每个元素进行操作，map和filter都有。map返回一个新数组，将元素进行处理后返回到新数组。

filter适合得筛选后才能返回出去的数组。foreach遍历数组，参数为一个回调函数，回调函数接收三个参数，当前元素，元素索引，整个数组， foreach 在对 item 进行修改的时候，如果 item 是原始类型的值，item 对应的 的内存地址实际并没有变化，如果 item 是 引用类型的值，item 对应多的内存地址也没有变化，但是对应的值，已经重写了。map与foreach类似，遍历数组，但其回调函数的返回值会组成一个新数组，新数组的索引结构和原数组一致，原数组不变；filter会返回原数组的一个子集，回调函数用于逻辑判断，返回true则将当前元素添加到返回数组中，否则排除当前元素，原数组不变。

**8.offsetWidth/offsetHeight、clienWidth/clienHeight与scrollWidth/scrollHeight的区别？**

1).OffsetWidth/offsetHeight返回值只包含content+padding+border，如有滚动条，也不包含滚动条

2).clientWidth/clientHeight返回值只包含content+padding，如有滚动条，也不包含滚动条

3).scrollWidth/scrollHeight返回值包含content+padding+溢出内容的尺寸

**9.==与===的区别？**

1).== 表示相等 （值相等）

2).===表示恒等（类型和值都要相等）

3).==只比较内容，而===既比较内容也比较数据类型，js在比较的时候如果是 == 会先做类型转换，再判断值得大小，如果是===类型和值必须都相等

**10.Null和undefined的区别？**

1).null是一个表示”无”的对象，转我数值是为0，undefined是一个表示”无”的原始值，转为数值时为NaN。当声明的变量还未被初始化时，能量的默认值为undefined

2).Null用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象

3).Undefined表示”缺少值”,就是此处应该有一个值,但是还没有定义。典型用法是

a.变量被声明了，但没有赋值时，就等于undefined

b.调用函数时，应该提供的参数没有提供，该参数等于undefined

c.对象没有赋值属性，该属性的值为undefined

d.函数没有返回值时，默认返回undefined

4).null表示”没有对象”，即该处不应该有值。典型用法是：

a.作为函数的参数，表示该函数的参数不是对象

b.作为对象原型链的终点

**11.字符串操作方法（8个）**

1).charCodeAt方法返回一个整数，代表指定位置字符的Unicode编码 //strObi.charCodeAt(index)

2).fromCharCode方法从一些Unicode字符串中返回一个字符串//string.fromCharCode([code1[code2...]])

3).charAt方法返回指定位置的字符 //strObj.CharAt(index)

4).slice方法返回字符串的片段 //strObj.slice(start[end])

start :要抽取的片断的起始下标。如果负数,则该参数规定的是从字符串的尾部开始算起的位置。也就是说，-1 指字符串的最后一个字符，-2指倒数第二个字符，以此类推。

end:紧接着要抽取的片段的结尾的下标。若未指定此参数，则要

提取的子串包括start到原字符串结尾的字符串。如果该参数负数,那么它规定的是从字符串的尾部开始算起的位置。

5).substring方法返回位于string对象中指定位置的子字符串 //strObj.substring(start,end)

start:必需。一个非负的整数,规定要提取的子串的第一个字符在stringObject中的位置。

stop :可选。-个非领的整数,比要提取的子串的最后一个字符在stringObject中的位置多1。如果省略该参数,那么返回的子串会一

直到字符串的结尾。

6).substr方法返回一个从指定位置开始的指定长度的子字符串//strObj.substr(start[length])

start:必需。要抽取的子串的起始下标。必须是数值。如果负数，那么该参数声明从字符串的尾部开始算起的位置。也就是说，-1指字符串中最后一个字符，-2 指倒数第二个字符，以此类推。

length:可选。子串中的字符数。必须是数值。如果省略了该参数，那么返回从stringObject的开始位置到结尾的字串。

7).indexOf方法返回string对象内第一次出现子字符串位置，如果没有找到子字符串，则返回1，indexOf方法对大小写敏感 //strObj.indexOf(substr[startindex])

8).lastIndexOf方法返回string对象中字符串最后出现的位置，如果没有匹配到子字符串，则返回1 //strObj.lastIndexOf(substr[startindex])

9).search方法用于检索字符串中指定的子字符串，或检索与正则表达式相匹配的子字符串。要执行忽略大写的检索，请追加标志i。如果没有找到任何匹配的子串，则返回-1。 //strObj.search(reExp)

10).concat方法返回字符串，该值包含了两个或多个提供的字符串的连接 //str.concat([string1,[string2...]])

11).toLowerCase方法返回一个字符串，该字符串中的字母被转换成小写

12).toLUpperCase方法返回一个字符串，该字符串中的字母被转换成大写

**12.数组操作方法（10个）**

1).shift()方法：把数组的第一个元素删除，并返回第一个元素的值

var arr=[1,2,3];  
console.log(arr.shift()); //[1]

Console.log(arr); //[2,3]

2).concat()方法：用于连接两个或多个数组,并返回一个新数组，新数组是将参数添加到原数组中构成的

var arr1=[1,2,3];

var arr2=[4,5]

var arr3=arr1.concat(arr2);

Console.log(arr1); //[1,2,3]

Console.log(arr); //[1,2,3,4,5]

3).join()方法：用于把数组中的所有元素放入一个字符串。元素是通过指定的，进行分隔的。返回一个字符串。该字符串是通过把 arrayObject 的每个元素转换为字符串，然后把这些字符串连接起来，在两个元素之间插入separator字符串而生成的。

var arr=[11,22];

Conlose.log(arr.join()); //2,3,4

Console.log(arr); //[2,3,4]

4).pop()方法：用于删除并返回数组的最后一个(删除元素)元素,如果数组为空则返回undefined ,把数组长度减 1

var arr=[2,3,4];

Console.log(arr.pop()); //4

Console.log(arr); //[2,3]

5).push()方法：可向数组的末尾添加一个或多个元素，并返回新的长度，（用来改变数组长度）

var a=[2,3,4]; var b=a.push(5);

Console.log(a); //[2,3,4,5]

Console.log(b); //4

6).reverse()方法用于颠倒数组中元素的顺序。

7).slice()方法：可从已有的数组中返回选定的元素。slice(开始截取位置，结束截取位置)

var arr=[2,3,4,5];

Console.log(arr.slice(1,3)); //[3,4]

Console.log(arr); //[2,3,4,5]

8).splice()方法向/从数组中添加/删除项目，然后返回被删除的项目。

splice()方法可删除从 index 处开始的零个或多个元素，并且用参数列表中声明的一个或多个值来替换那些被删除的元素。

var arr =[5,6,7,8];

Console.log(arr.splice(1,0,9)); //[]

Console.log(arr); //[5,9,6,7,8]

Var b=[5,6,7,8];

Console.log(b.splice(1,2,3)); //[6,7]

Console.log(b); //[5,3,8]

9).substring()和substr()相同点：如果只是写一个参数，两者的作用都有一样，都是截取字符串从当前下标直到字符串最后的字符串片段。

substr(startIndex);

substring(startIndex);

var str=’123456789’;

Console.log(str.substr(2)); //’3456789’

Console.log(str.substring(2)); //’345679’

不同点：第二个参数。substr(startIndex,length);第二个参数是截取字符串的长度（从起始点截取某个长度的字符串）。

Substring(startIndex,endIndex);第二个参数是截取字符串最终的下标（截取2个位置之间的字符串，‘含头不含尾’）。

Console.log(‘123456789’,substr(2,5)); //’34567’

Console.log(‘123456789’,substring(2,5)); //’345’

10).sort()排序，按指定的参数对数组进行排序 默认升序。

var arr =["cherries","apples","bananas"];

arr.sort(); //[‘apples’,’bananas’,’cherries’];

var scores =[1,10,21,2;

scores.sort(); //[1,10,2,21]

**13.JS中定时器有哪些？它们的区别及用法是什么？**

1).setTime 只执行一次；

2).SetInterval 会一直重复执行

**14.Document.write和innerHTML的区别？**

主要区别：document.write是直接将内容写入页面的内容流，会导致页面全部重绘，innerHTML将内容写入某个DOM节点，不会导致页面全部重绘

1).write是DOM方法,向文档写入HTML表达式或JavaScript代码，可列出多个参数，参数被顺序添加到文档中 ；innerHTML是DOM属性,设置或返回调用元素开始结束标签之间的HTML元素。  
 2).两者都可向页面输出内容,innerHTML比document.write更灵活。  
 当文档加载时调用document.write直接向页面输出内容，文档加载结束后调用document.write输出内容会重写整个页面。通常按照两种的方式使用 write() 方法：一是在使用该方在文档中输出 HTML，二是在调用该方法的的窗口之外的窗口、框架中产生新文档（务必使用close关闭文档）。  
 在读模式下，innerHTML属性返回与调用元素的所有子节点对应的HTML标记，在写模式下，innerHTML会根据指定的值创建新的DOM树替换调用元素原先的所有子节点。  
 3).两者都可动态包含外部资源如JavaScript文件  
通过document.write插入<script>元素会自动执行其中的脚本；  
大多数浏览器中，通过innerHTML插入<script>元素并不会执行其中的脚本。

**15.解释一下DOM和BOM？**

1).DOM 全称是 Document Object Model，也就是文档对象模型。是针对XML的基于树的API。描述了处理网页内容的方法和接口，是HTML和XML的API，DOM把整个页面规划成由节点层级构成的文档。

这个DOM定义了一个HTMLDocument和HTMLElement做为这种实现的基础,就是说为了能以编程的方法操作这个 HTML 的内容（比如添加某些元素、修改元素的内容、删除某些元素），我们把这个 HTML 看做一个对象树（DOM树），它本身和里面的所有东西比如 <div></div> 这些标签都看做一个对象，每个对象都叫做一个节点（node），节点可以理解为 DOM 中所有 Object 的父类。

2).BOM 是 Browser Object Model，浏览器对象模型。刚才说过 DOM 是为了操作文档出现的接口，那 BOM 顾名思义其实就是为了控制浏览器的行为而出现的接口。

浏览器可以做什么呢？比如跳转到另一个页面、前进、后退等等，程序还可能需要获取屏幕的大小之类的参数。所以 BOM 就是为了解决这些事情出现的接口。比如我们要让浏览器跳转到另一个页面，只需要location.href = "http://www.xxxx.com";这个 location 就是 BOM 里的一个对象。

BOM的核心是Window，而Window对象又具有双重角色，它既是通过js访问浏览器窗口的一个接口，又是一个Global（全局）对象。这意味着在网页中定义的任何对象，变量和函数，都以window作为其global对象。Window对象包含属性：document、location、navigator、screen、history、frames、Document根节点包含子节点：forms、embeds、anchors、images、links从window.document已然可以看出，DOM的最根本的对象是BOM的window对象的子对象。

由于BOM的window包含了document，因此可以直接使用window对象的document属性，通过document属性就可以访问、检索、修改XHTML文档内容与结构。因为document对象又是DOM（Document Object Model）模型的根节点。

可以说，BOM包含了DOM(对象)，浏览器提供出来给予访问的是BOM对象，从BOM对象再访问到DOM对象，从而js可以操作浏览器以及浏览器读取到的文档。

**16.jQuery一个对象可以同时绑定多个事件，这是如何实现的？**

bind on delegate live进行多事件绑定的原理

**17.针对jQuery的优化方法？**

1).缓存频繁操作DOM对象

2).尽量使用ID选择器代替类选择器

3).总是从#ID选择器来继承

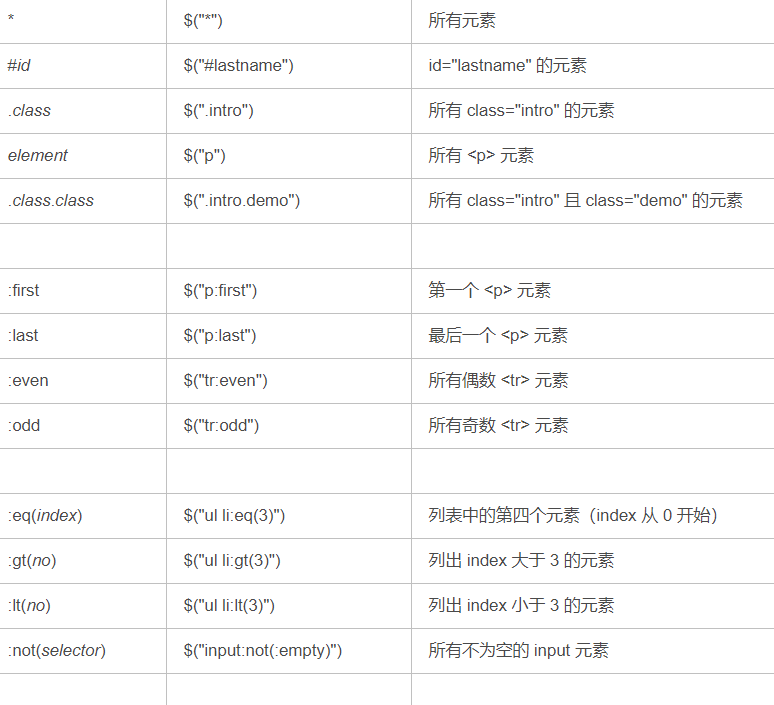
4).尽量使用链式操作

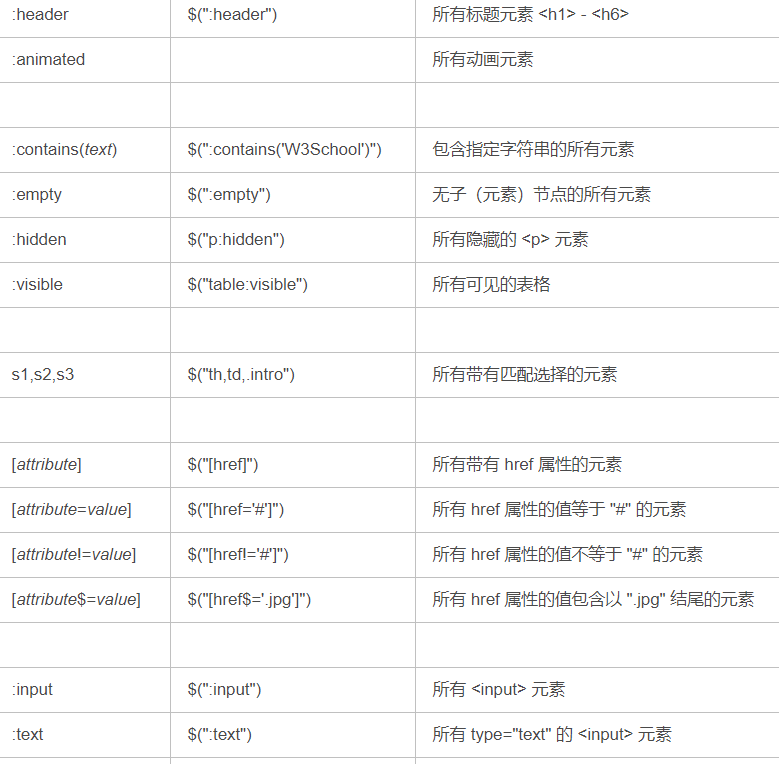
5).在绑定事件上使用时间委托

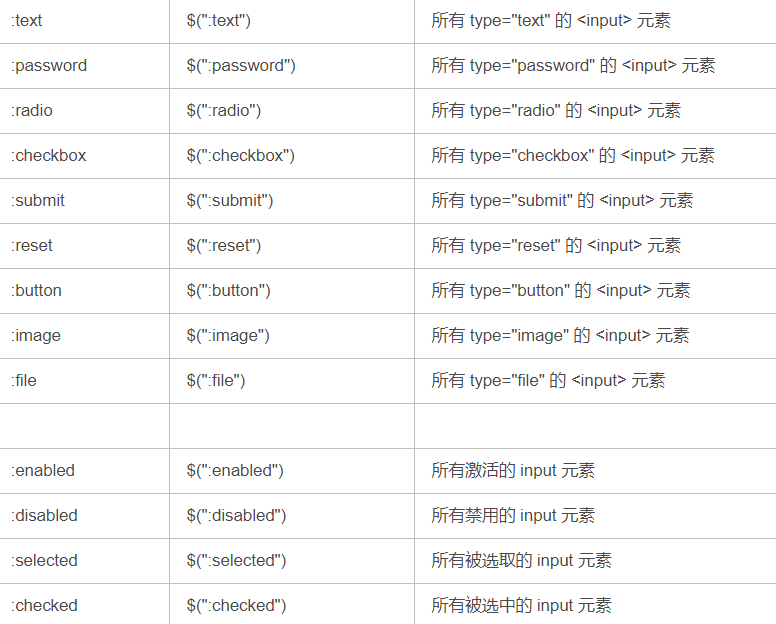
6).采用的jQuery的内部函数数据来储存

7).使用最新版本的jQuery

**18.jQuery常用选择器**







1、类选择器HTML代码：<div class="mydiv"></div>Jquery获取对象：$(".mydiv")

2、ID选择器HTML代码：<div id="mydiv"></div>Jquery获取对象：$("#mydiv")

3、标签选择器HTML代码：<span></span>Jquery获取对象：$("span")

4、通配符选择器HTML代码：<div><span>文字</span><h1>标题</h1><p>段落</p></div>Jquery获取对象：$("div \*")，获取div下的所有标签对象

5、同时定义多个选择器HTML代码：<div id="mydiv"></div><span class="other"></span>Jquery获取对象：$("#mydiv,.other")，注意有逗号隔开了

6、子选择器HTML代码：<h1><span></span></h1>Jquery获取对象：$("h1 > span")，需要注意的是，这种方式只会获取H1下面的SPAN，如果H1的孙子还有SPAN，将不会进行查找，这种写法仅用于查找子这一代

7、下一个兄弟选择器HTML代码：<h1></h1><span>1</span><span>2</span>Jquery获取对象：$("h1").next("span")，该用法只会查找到<span>1</span>的元素，即H1的下一个兄弟

8、同级选择器HTML代码：<h1></h1><span></span><p></p><a></a>Jquery获取对象：$("h1").nextAll()，获取h1的所有同级对象

9、选取第一个和最后一个选择器HTML代码：<p>段落1</p><p>段落2</p><p>段落3</p>Jquery获取对象：$("p:first")获取第一个P标签，$("p:last")获取最后一个P标签

10、取非选择器HTML代码：<div class="mydiv"></div><div class="demo"></div>Jquery获取对象：$("div:not(.mydiv)")，只会选择class为demo的对象。

基本选择器：

　　基本选择器可以分为三种,分别为：标签选择器、类选择器、ID选择器、并集选择器、交集选择器和全局选择器。

层次选择器：

　　层次选择器有四种，分别为：后代选择器、子选择器、相邻选择器和同辈选择器，常用的是后代选择器和子选择器。

属性选择器：

用于选择包含给定属性的所有元素。

　　格式：$("selector[attribute]")

　　　例：　$("div[id]")

　　　　　　从文档中选择包含id属性的div元素。

过滤选择器：

　　过滤选择器可以使用元素的索引值，内容，属性，子元素位置，表单域属性以及可见性作为筛选条件获取相关元素。

基本过滤选择器：

　　根据索引值对元素进行筛选，类似于CSS的伪类选择器，以冒号（：）开头；并且要和另一选择器一起使用。

可见性过滤选择器：

　　jQuery选择器出了可以通过CSS选择器、位置选取元素等，还能通过元素的显示状态，即元素显示或隐藏来选取元素。

　　：visible：选取所有可见的元素。

　　：hidden：选取所有隐藏的元素。

**jQuery重点注意事项**

　　1）选择器中包含特殊符号

　　2）选择器中含有空格

**四.商城开发过程中遇到的问题及解决方案？**

**1.HTML问题**

1).审查问题

2).注释（删除）排错

3).曾在服装板块布局有很大的问题 导致后面楼层跳转出错

2.Css

1).检查元素和样式

2).有样式无效果，内联布局。用网页F12调试器解决好了

3).样式范围过大，盒子之间的间隙不匹配

ul.myul

ul li>p(img) 子父选择器，兄弟选择器，相邻选择器。

ul>li\*6（ul下6个li标签）

#menu ul {list-style:none;margin:0px;}  
list-style:none，这一句是取消列表前点，因为我们不需要这些点。  
margin:0px，这一句是删除UL的缩进，这样做可以使所有的列表内容都不缩进。

**3.注意事项**

1).css选择器优先级：!important>行内>id>类>标签

!important > 行内样式 > ID选择器 > 类选择器 > 标签 > 通配符 > 继承 > 浏览器默认属性。

css选择器优先级最高到最低顺序为：  
  
1.id选择器（#myid）  
  
2.类选择器（.myclassname）  
  
3.标签选择器（div,h1,p）  
  
4.子选择器（ul < li）  
  
5.后代选择器（li a）  
  
6.伪类选择（a:hover,li:nth-child）  
  
需要注意的是：　　  
  
!important的优先级是最高的，但出现冲突时则需比较”四位数“;  
优先级相同时，则采用就近原则，选择最后出现的样式;  
继承得来的属性，其优先级最低。

2).css有哪些属性可以继承，哪些不可以？

不可以继承：

display：规定元素应该生成的框的类型  
  
text-decoration：规定添加到文本的装饰  
  
text-shadow：文本阴影效果  
  
white-space：空白符的处理  
  
盒子模型的属性：width、height、margin 、border、padding  
  
背景属性：background  
  
定位属性：float、clear、position、top、right、bottom、left、min-width、min-height、max-width、max-height、overflow、clip、z-index

可以继承：

font：组合字体  
  
font-family：规定元素的字体系列  
  
font-weight：设置字体的粗细  
  
font-size：设置字体的尺寸  
  
font-style：定义字体的风格  
  
text-indent：文本缩进  
  
text-align：文本水平对齐  
  
line-height：行高  
  
color：文本颜色  
  
visibility：元素可见性  
  
光标属性：cursor  
  
所有元素可以继承的  
  
1、元素可见性：visibility  
  
2、光标属性：cursor  
  
内联元素可以继承的属性  
  
1、字体系列属性  
  
2、除text-indent、text-align之外的文本系列属性  
  
块级元素可以继承的属性  
  
text-indent、text-align  
  
inherit（继承）值  
  
每一个属性可以指定值为“inherit”，即：对于给定的元素，该属性和它父元素相对属性的计算值取一样的值。继承值通常只用作后备值，它可以通过显式地指定“inherit”而得到加强，例如：  
  
p { font-size: inherit; }  
  
继承的局限性  
  
继承虽然减少了重复定义的麻烦，但是，有些属性是不能继承的，例如border（边框）、margin（边距）、padding（补白）和背景等。  
  
这样设定是有道理的，例如，为一个

设定了边框，如果此属性也继承的话，那么在这个

内所有的元素都会有边框，这无疑会产生一个让人眼花缭乱的结果。同样的，影响元素位置的属性，例如margin（边距）和padding（补白），也不会被继承。

3).水平和垂直居中（三种方式）

4).清除浮动的三种方式

a.text-align是让里面的内容水平居中

b.line-height是行高，行高等于元素的高度，就能让内容垂直居中

d.left和top 50%是根据HTML来定位的，只能让元素的左上顶点

5).position有几个值？代表什么？

a.absolute生成绝对定位的元素，相对于 static 定位以外的第一个父元素进行定位。元素的位置通过 "left", "top", "right" 以及 "bottom" 属性进行规定。

b.relative生成相对定位的元素，相对于其正常位置进行定位。

因此，"left:20" 会向元素的 left 位置添加 20 像素

c.fixed生成绝对定位的元素，相对于浏览器窗口进行定位。元素的位置通过 "left", "top", "right" 以及 "bottom" 属性进行规定。

7).box-sixing有几值？代表什么？

a.content-box：默认标准盒型,width+padding+border+margin  
b.border-box：IE标准，怪异盒模型，总宽width+margin  
c.inherit：从父元素继承box-sizing属性

**4.JS如何解决异步问题**

Promise对象解决异步问题

将回调变为链式调用，让代码更易读

resolve方法，将状态变更为成功

catch方法，可以捕获异常。

这就是我的一个项目总结报告。