进销存管理系统实训课件

主讲：汤燕

* 收齐需求分析文档
* Ajax加载商品类别
* 断点调试讲解
* 二维码生成技术
* 自己完成退换货模块

# Ajax加载商品类别

## 前端代码

|  |
| --- |
| $(**function**(){  $.post("./../ajax?" + Math.random(), { rnum: "5" }, **function** (res) {  $("#typeidbox").html(res.msg);  });    }); |

## 后端代码

|  |
| --- |
| /\*  获取商品类别html字符串for添加页面  \*/  **private List<tbtype> listall = new ArrayList<tbtype>();**  **protected** **void** getprotypehtml(HttpServletRequest request,HttpServletResponse response) **throws** IOException{  listall.clear();  getData("0");  String html="<select name='typeid'>";  **for**(tbtype item:listall)  {  String spacehtml="|--";  **for**(**int** i=1;i<=item.getlevelnum();i++)  {  spacehtml+="|--";  }    html+="<option value='"+item.getId()+"'>"+spacehtml+item.getTypeName()+"</option>";  }  html+="</select>";  response.setCharacterEncoding("utf-8");  response.setContentType("text/json;charset=utf-8");  response.getWriter().write("{\"msg\":\""+html+"\"}");  } |

# 二维码生成技术



|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*生成二维码图片开始\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  String twocodeimgPath = **this**.getServletContext().getRealPath("/twocode");  String twocodefilename=getDateFormat()+".jpg";  **try** {  String content =pronum;  MultiFormatWriter multiFormatWriter = **new** MultiFormatWriter();  Map hints = **new** HashMap();  hints.put(EncodeHintType.*CHARACTER\_SET*, "UTF-8");  BitMatrix bitMatrix = multiFormatWriter.encode(content, BarcodeFormat.*QR\_CODE*, 400, 400,hints);  File file1 = **new** File(twocodeimgPath,twocodefilename);  MatrixToImageWriter.*writeToFile*(bitMatrix, "jpg", file1);  } **catch** (Exception e) {  e.printStackTrace();  }    /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*生成二维码图片结束\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ |

# 代码调试

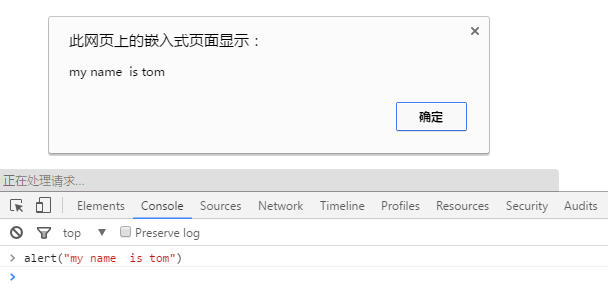
## 为什么掌握这些调试技巧

调试技巧，在任何一项技术研发中都可谓是必不可少的技能。掌握各种调试技巧，必定能在工作中起到事半功倍的效果。譬如，快速定位问题、降低故障概率、帮助分析逻辑错误等等。而在互联网前端开发越来越重要的今天，如何在前端开发中降低开发成本，提升工作效率，掌握前端开发调试技巧尤为重要。

这些前端JS调试技巧，也许你已经熟练掌握，那让我们一起来温习，也许有你没见过的方法，不妨一起来学习，也许你尚不知如何调试，赶紧趁此机会填补空白。

## 骨灰级调试Alert

那还是互联网刚刚起步的时代，网页前端还主要以内容展示为主，浏览器脚本还只能为页面提供非常简单的辅助功能的时候。那个时候，网页主要运行在以IE6为主的浏览器中，JS的调试功能还非常弱，只能通过内置于Window对象中的alert方法来调试，那时候看起来应该是这个样子：



需要说明一点，当年还没有这么高级的控制台，而alert的使用也是在真实的页面JS代码中。alert的调试方式很原始，但当时确实有它不可磨灭的价值，甚至到今天，已然有其用武之地。

缺点：

1. 弹出模态的调试信息窗口，不能操作其他地方。
2. 窗口着实不太美观，而且会遮挡部分页面内容。
3. 窗口会阻塞程序继续执行，要确定之后才进行下一步。

## 新一代调试王者Console

随着JS在Web前端中能做的事情越来越多，责任越来越大，而地位也越来越重要。传统的alert调试方式已经渐渐不能满足前端开发的种种场景。

### 常用的方法

**console.log(object[, object, ...])，使用频率最高的一条语句**：向控制台输出一条消息。支持 C 语言 printf 式的格式化输出。

console.info(object[, object, ...]) ，向控制台输出一条信息，该信息包含一个表示“信息”的图标，和指向该行代码位置的超链接。

console.error(object[, object, ...])在控制台中输出一条错误信息。

console.warn(object[, object, ...])在控制台中输出一条警告信息。

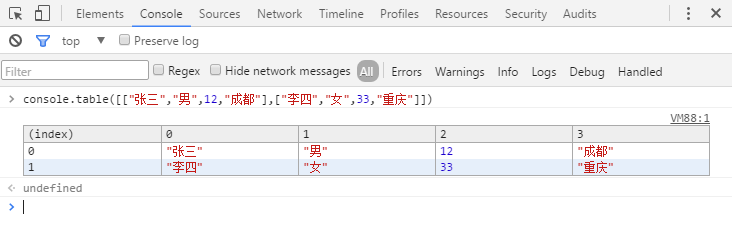
**console.dir(object) 输出一个对象的全部属性**（输出结果类似于 DOM 面板中的样式）。

**console.clear() 清空控制台。**

以Chrome浏览器为首的后起之秀，为Console扩展了更丰富的功能：

### 输出表格

例1： console.table([["张三","男",12,"成都"],["李四","女",33,"重庆"]])



### 美化console.log的文本

先看两个效果

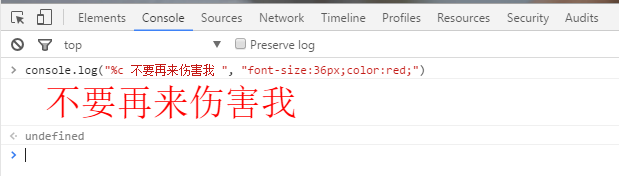
1、打开百度按F12，就会出现下图效果：



再看支付宝：



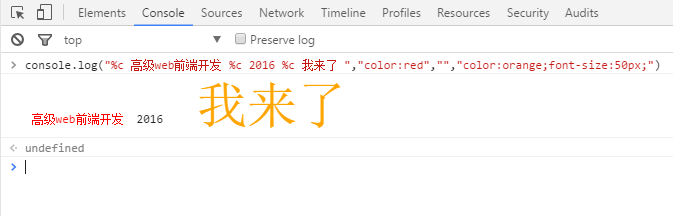
例1：一行文本单个样式。



console.log("%c 不要再来伤害我 ", "font-size:36px;color:red;")

例2：一行文本多个样式，每个%c 对应一个样式

console.log("%c 高级web前端开发 %c 2016 %c 我来了 ","color:red","","color:orange;font-size:50px;")



Console.log输出文字样式支持css3

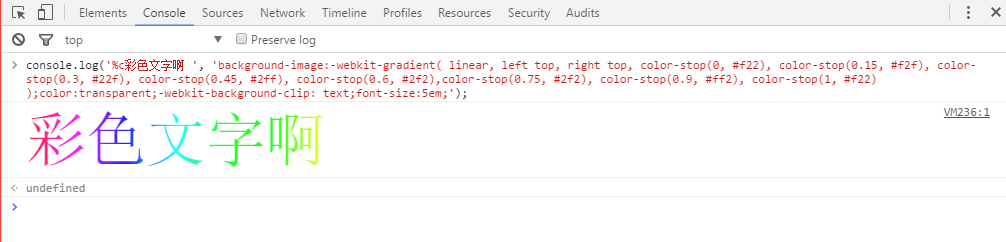
例3：给文字加阴影。

console.log("%c两情若是久长时，又岂在朝朝暮暮！", "text-shadow: 3px 1px 1px grey;font-size:36px");



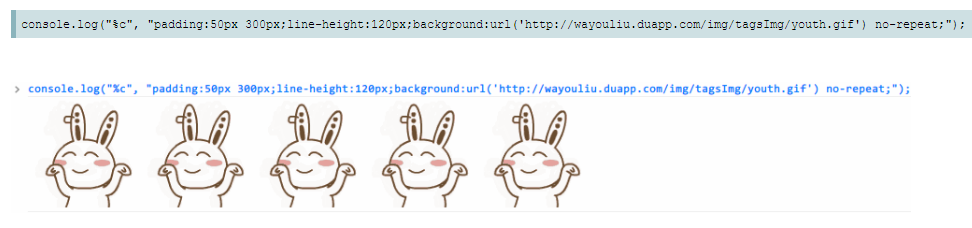
例4：彩色文字。

console.log('%c彩色文字啊 ', 'background-image:-webkit-gradient( linear, left top, right top, color-stop(0, #f22), color-stop(0.15, #f2f), color-stop(0.3, #22f), color-stop(0.45, #2ff), color-stop(0.6, #2f2),color-stop(0.75, #2f2), color-stop(0.9, #ff2), color-stop(1, #f22) );color:transparent;-webkit-background-clip: text;font-size:5em;');



例5：输出背景图片，甚至是gif动画作背景图片。

console.log("%c", "padding:50px 300px;line-height:120px;background:url('http://wayouliu.duapp.com/img/tagsImg/youth.gif') no-repeat;");



### JS断点调试

JS断点调试，即是在浏览器开发者工具中为JS代码添加断点，让JS执行到某一特定位置停住，方便开发者对该处代码段的分析与逻辑处理。为了能够观察到断点调试的效果，我们预先随意准备一段JS代码：

例如，最终想输出m5,结果把m5写成了m2,传统调试可以在代码里加很多个alert,但是这样调试效率低，老弹出确认框，此时就可以打断点调试。

<script type="text/javascript">

function add(a,b)

{

var c=a+b

return c;

}

$(function(){

$("#btn1").click(function(){

var m1=100;

var m2=m1+200;

var m3=m1-m2;

var m4=add(m1,m3);

var m5=m4;

$("#box1").text(m5);

console.log(m5);//例如m5不小心写成m2了，就可以打断点调试

});

});

</script>

# 面试相关

要学习的技术非常多，但是要抓住重点，建议大家后面重点关注以下技术：

H5 css3 ajax node angularjs react vue.js

## 准备内容

查看**招聘网站**，他们要求什么，我们就学什么。

简历格式doc ,不要pdf.

## 工资范围

范围跨度不要太大，比如6000-9000，明显跨度太大，人家给你6000还是9000呢。

## 面试地点

不要选择宾馆、会所等隐蔽的地点面试。

异地面试具有极大的风险性。

**可以选择视频面试、电话面试。**

**最好是到公司里面试，可以更加全面的了解公司。**

## 性别歧视

在招聘需求中，应尽量看清要求，例如对年龄、性别的要求，避免错投简历。当然，若求职者遇到“性别歧视”的情况，可以拿起法律武器，依法维护自己的权益。

您真的是来面试高级前端工程师的吗？

有的人想通过面试寻找问题的解决方案，让求职者给出解决方案。

## 换位思考，假如你是HR

也许你觉得找个工作不容易，其实用人单位想招聘一个满意的人才也不容易。

有的人是入职当天就离职。

第一步就是要分析想要什么样的人，不管是个人喜好，还是团队气质，技术水平，甚至是性别年龄，都是一些衡量标准，把这些标准用优先级列一个表格出来，按照表格继续进行下一步的分析。

技术热情很重要，按时完成任务更重要。

做事态度很重要，不想招聘天天抱怨却不去解决问题的人。

学习能力，总结能力，能够自我进步的人，而不是需要手把手教的。

分析能力，其实也就是解决问题的能力。

技术经验，当然越多越好，如果缺乏，至少要表现出成长潜力。

## 面试题

自我介绍。

为什么离开上家公司。有的公司还要求开离职证明。

1. 你都做过哪些项目？

企业网站、商城、进销存系统、微信小程序、webApp。

1. 在项目中主要负责什么？

比如做商城的时候，主要负责商城界面的重构，动画的制作，页面的优化，插件的开发等。

3、遇到过哪些技术难题，如何解决的？

样式冲突，css兼容性问题, js兼容性问题。

解决办法：css冲突主要通过审查元素，查看哪些样式被重写了，重新给这些标签定义id或者class,重新定义样式。js尽量使用jquery，因为jquery兼容性做得比较好。通过控制台打印输出，或者断点调试。

1. IE6兼容性问题是怎么处理的？

我们公司早就停止兼容IE6了，如果确实要做兼容可以 采用hack技术或者 条件注释 解决。

5、 $(this) 和 this 关键字在 jQuery 中有何不同？

$(this) 返回一个 jQuery 对象，你可以对它调用多个 jQuery 方法，比如用 text() 获取文本，用val() 获取值等等。而 this 代表当前元素，它是 JavaScript 关键词中的一个，表示上下文中的当前 DOM 元素。

1. 使用 CDN 加载 jQuery 库的主要优势是什么？

1、节省自己服务器带宽。

2、更快的下载速度。

3、如果浏览器已经从同一个CDN下载类相同的 jQuery 版本, 那么它就不会再去下载它一次。

你对我们公司了解多少？

我们产品的改进意见或者建议？

职业规划？

打算深挖前端，把技术提高到一定的高度，有机会再带带团队。

1. 完成老师上课已经开发过的功能模块。
2. 自己必须要有一个创新模块。

比如自己多做几张报表，增加供应商管理模块、增加经销商管理模块。

增加配送驾驶员管理模块。

美化界面等。

1. 上传实训日志。
2. 上交实训需求分析文档。
3. 完成项目测试。
4. 编写软件帮助手册。