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->JQUERY->day04

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->JQUERY->day04

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->JQUERY->day04

16. jquery 跨域 cors ... express脚手架

作业: 高频笔试题 jsonp 跨域

3. 单词列表: 小程序->单词->JQUERY->day04

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. 添加自定义函数

2. 封装自定义插件

3. ajax

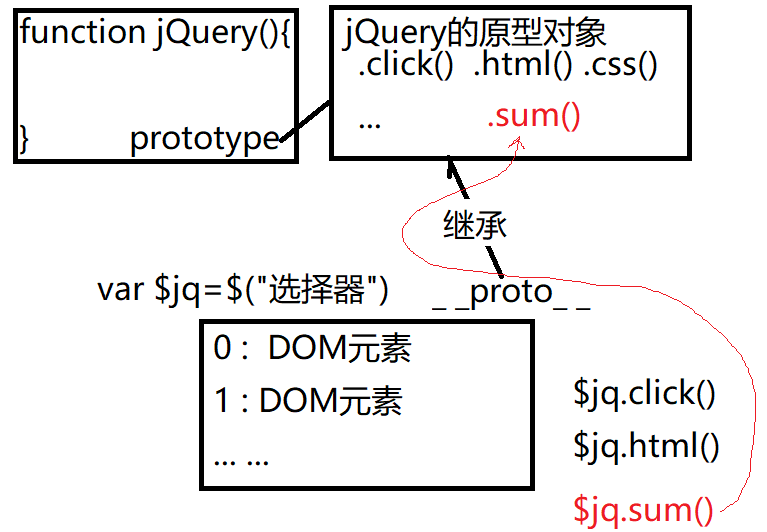
4. \*\*\*跨域

对应小程序视频: 小程序->在线->JQUERY->day04: 14 添加jQUery自定义函数...

一. 添加自定义函数:

1. 如果我们经常使用的一个功能，jQuery中没有提供，其实，我们也可以自定义函数，加入到jquery中

2. 如何: 只需要定义一个函数，放入jQuery的原型对象中。所有jquery家孩子都可以使用这个自定义的函数



3. 示例: 为jquery添加sum()函数，可以对找到的所有DOM元素求和

11\_sum.html

|  |
| --- |
| <!DOCTYPE html>  <html>  <head lang="en">    <meta charset="UTF-8">    <title></title>  </head>  <body>    <ul>      <li>85</li>      <li>91</li>      <li>73</li>      <li>59</li>    </ul>    <script src="js/jquery-1.11.3.js"></script>    <script>      //为jQuery家原型对象中添加一个sum()函数      //jQuery.prototype.sum=function(){      //**jQuery.fn=jQuery.prototype**      jQuery.fn.sum=function(){        console.log(`调用一次jQuery原型对象中我们自定义的sum()函数...`);        //对一个数组中的内容求和的固定套路:        //1. 自定义一个变量准备保存累加的和        var result=0;        //2. 遍历数组中每个元素        //问题: 想对找到的所有DOM元素内容求和，应该遍历谁?        //答: 应该遍历本次$("选择器")找到的查找结果对象。因为其中包含了找到的所有DOM元素        //问题: **如何获得将来调用sum()函数的.前的$("选择器")查找结果对象？**        //答: 只有一种办法: **this**        //问题: this是否需要$()包裹？        //答: 不用。**因为.前已经是$("选择器")的一个jq对象了。所以，不需要再重复$()**        for(var i=0;i<this.length;i++){          //3. 每遍历数组中一个元素，就将元素值累加到自定义变量result上          //错误:**this[i]取出的是jq查找结果中某一个DOM元素对象，而不是元素的内容**          //result+=this[i]          //正确:          //先获得当前查找结果中i位置的元素对象          //再元素对象.innerHTML获得元素的内容          //保险起见，将字符串类型的元素内容转为数字          //parseFloat既可转整数，又可转小数          //复习第一阶段parseFloat和parseInt          result+=parseFloat(this[i].innerHTML)        }        //当循环结束后，自定义变量result上就累加了所有元素的值        //4. 返回求和的结果        return result;      }        //比如: 经常需要对找到的所有DOM元素的内容求和      //问题: jq中没有提供对元素内容求和的函数      console.log($("ul>li").sum());      //          查找结果      //          包含了找到的所有DOM元素对象    </script>  </body>  </html>  运行结果: |

二. 封装自定义插件:

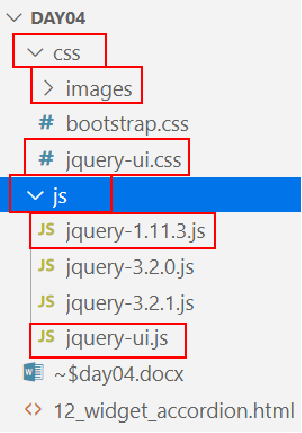
1. 什么是插件: **包含专属的HTML+JS+CSS的可重用的独立的页面功能区域**

2. 为什么: 重用——公司极其重视重用！

3. 何时: 今后只要出现一个功能需要多处反复使用时，都要先将功能封装为插件，再反复使用插件。

4. jQuery官方提供了一套插件库: jQuery UI

(1). 官网: jqueryui.com



(2). 在页面中引入jqueryui的css和js

<script src="js/jquery-1.11.3.js">

<script src="js/jquery-ui.js">

<link rel="stylesheet" href="css/jquery-ui.css">

(3). 如何使用:

a. 先在页面中按插件的要求编写HTML内容

b. 再在自定义js中，用jquery查找到插件的父元素，调用jqueryui提供的插件函数

5. 示例: 使用jqueryui实现手风琴效果:

12\_widget\_accordion.html

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title> new document </title>    <meta charset="utf-8">    <script src="js/jquery-1.11.3.js"></script>    <script src="js/jquery-ui.js"></script>    <link rel="stylesheet" href="css/jquery-ui.css">   </head>   <body>    <h1>jQueryUI：Widgets —— Accordion</h1>    <!--复制14\_accordion.html中<div class="accordion">的内容到此-->    <div id="my-accordion">      <div>《西游记》简介</div>      <div>一个和尚和四个动物的故事: Lorem ipsum dolor sit amet, consectetur adipisicing elit. Similique nulla voluptas velit minus esse voluptatem illum quis magni nihil sint facilis cupiditate nobis quia ab neque. Modi veniam omnis nisi? </div>      <div>《水浒传》简介</div>      <div>105个男人和三个女人的故事: Lorem ipsum dolor sit amet, consectetur adipisicing elit. Omnis provident sapiente aperiam reprehenderit repellat rem magnam vel odio quia harum hic impedit dolorem similique ea est consequatur adipisci at nemo!</div>      <div>《红楼梦》简介</div>      <div>一个男人和一群女人的故事: Lorem ipsum dolor sit amet, consectetur adipisicing elit. Delectus minima quidem aspernatur eligendi optio cupiditate minus nam expedita? Aliquid veritatis doloribus maxime vel dicta illo unde iusto qui quasi doloremque.</div>    </div>    <script>      $("#my-accordion").accordion();    </script>   </body>  </html>  运行结果: |

6. jquery ui 2个问题:

(1). 只有PC端，没有移动端！

(2). 都是在js代码中写死的，极其不便于维护！

7. 但是jquery ui这种简化的思路，如果用在我们自己的项目中还是非常有好处的！所以，如果我们自己封装插件给自己用时，还是非常建议参考jquery ui这种方式！

8. jquery ui封装插件的方式:

(1). jquery ui所有插件，其实都是一个jquery的自定义函数，保存在jquery的原型对象中。所以，任何一个jquery查找结果对象都可调用插件函数

(2). 问题: 页面上一个class都没有，css样式怎么添加到元素上的？

答: 当我们调用**jquery插件函数**时，jquery插件函数会根据自身的需要，**自动**给插件的元素及其子元素**添加class**，不再需要使用插件的人手工添加任何class

(3). 问题: 我们没有写任何事件绑定的代码，但是只调用插件函数，插件中的元素就能点击了！

答；当我们调用**jquery插件函数**时，jquery插件函数会根据自身的需要，**自动**给插件中对应的元素**绑定事件处理函数**。不再需要插件的使用者手工添加事件绑定代码。

9. 如何自己封装一个jQuery自定义插件?

(0). 前提: 封装插件的过程不是从0开始开发的过程，而是**将一个已经用传统的HTML+CSS+JS实现了的界面效果**，**提取**成一个插件而已。所以，要封装插件，**必须已经用传统的HTML+CSS+JS实现了界面效果**

(1). 将插件涉及的**css代码提取**到一个专门的**独立的css文件**中保存

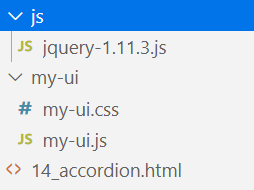
(2). 在一个专门的**独立的js文件中**，为jquery**原型对象添加**一个**自定义**的插件函数，做两件事:

**a. 自动为插件元素及其子元素添加class**

**b. 为插件中指定元素添加事件绑定**

10. 如何使用自己封装好的插件: 同使用jquery ui插件完全相同！

11. 示例: 封装手风琴效果的自定义插件



my-ui/my-ui.css

|  |
| --- |
| .accordion{width:80%; margin:0 auto;}  .accordion>.title{    background:#eee; border:1px soild #aaa;    padding:6px; font-size:1.5em;    font-weight:bold; cursor:pointer;  }  .accordion>.content{    border-left:1px solid #eee;    border-right:1px solid #eee;  }  .accordion>:last-child{    border-bottom:1px solid #eee;  }  .fade{    height:0;    opacity:0;    overflow:hidden;    transition:all .5s linear;  }  .in{    height:84px;    opacity:1;  } |

my-ui/my-ui.js

|  |
| --- |
| //为jquery的原型对象添加一个自定义插件函数myAccordion()，希望将来用户一调用这个myAccordion()插件函数，class和事件绑定就自动加入到插件元素上。  jQuery.fn.myAccordion=function(){    //this->将来调用这个插件函数的.前的jquery对象，也就是整个插件的父元素对象。    //起个别名    var $parent=this;    //自动做2件事:    //1. 为元素自动添加class      //1.1 给父元素自己加class accordion    $parent.addClass("accordion")    //1.2 给父元素下所有奇数位置的直接子元素加class title    .children(":nth-child(odd)") //css中下标从1开始    .addClass("title") //return 奇数位置的子元素    //1.3 给所以奇数位置的子元素的下一个兄弟加class content fade    .next().addClass("content fade") //return 所有奇数位置的子元素的下一个兄弟都返回回来    //1.4 给所有奇数位置的子元素的下一个兄弟中第一个div加class in    //first()可获得当前查询结果中第一个位置的元素，且自动包装为jq对象。.first()=>$($(查找结果)[0])    .first().addClass("in");    //2. 为元素自动绑定事件处理函数    //直接将页面中已经实现的事件绑定代码剪切到插件函数中，由插件函数负责自动执行事件绑定操作    $(".accordion").on("click",".title",e=>      $(e.target).next(".content").toggleClass("in")        .siblings(".content").removeClass("in")    );  }  //将来myAccordion()函数会这样调用：  //$("#my-accordion").myAccordion()  //                       this //所以this不用$()包裹  //                         |  //$(整个插件的父元素) ←-点前--  //已经是$(...)了！ |

14\_accordion.html

|  |
| --- |
| <!DOCTYPE html>  <html>   <head>    <title> new document </title>    <meta charset="utf-8">    <script src="js/jquery-1.11.3.js"></script>    <script src="my-ui/my-ui.js"></script>    <link rel="stylesheet" href="my-ui/my-ui.css">   </head>   <body>    <h1>使用“高度动画”实现“手风琴”组件</h1>    <div id="my-accordion">      <div>《西游记》简介</div>      <div>一个和尚和四个动物的故事: Lorem ipsum dolor sit amet, consectetur adipisicing elit. Similique nulla voluptas velit minus esse voluptatem illum quis magni nihil sint facilis cupiditate nobis quia ab neque. Modi veniam omnis nisi? </div>      <div>《水浒传》简介</div>      <div>105个男人和三个女人的故事: Lorem ipsum dolor sit amet, consectetur adipisicing elit. Omnis provident sapiente aperiam reprehenderit repellat rem magnam vel odio quia harum hic impedit dolorem similique ea est consequatur adipisci at nemo!</div>      <div>《红楼梦》简介</div>      <div>一个男人和一群女人的故事: Lorem ipsum dolor sit amet, consectetur adipisicing elit. Delectus minima quidem aspernatur eligendi optio cupiditate minus nam expedita? Aliquid veritatis doloribus maxime vel dicta illo unde iusto qui quasi doloremque.</div>    </div>    <script>      $("#my-accordion").myAccordion();    </script>   </body>  </html>  运行结果: |

三. ajax

对应小程序视频: 小程序->在线->JQUERY->day04 15 jQuery发送ajax请求$.ajax

1. 从jquery开始，今后几乎所有的函数库或框架都提供了专门发送ajax请求的函数。从此，任何时候发送ajax请求，只需要一句话！

2. jquery中发送get和post类型的ajax请求: (固定写法！)

**$.ajax({**

**url:"服务器端接口地址",**

**type: "get或post",**

**data:{ 参数名:参数值, ... ... },**

**dataType:"json"** , //说明服务器端返回的是json字符串，需要自动调用JSON.parse()转为js对象或数组

**success:function( result ){** //像第二极端的onreadystatechange，只在请求响应成功结束后才自动执行！

//result就是服务器端返回的响应结果，且已经被JSON.parse()转为了js的数组或对象，不再需要手工调用JSON.parse()了！

**}**

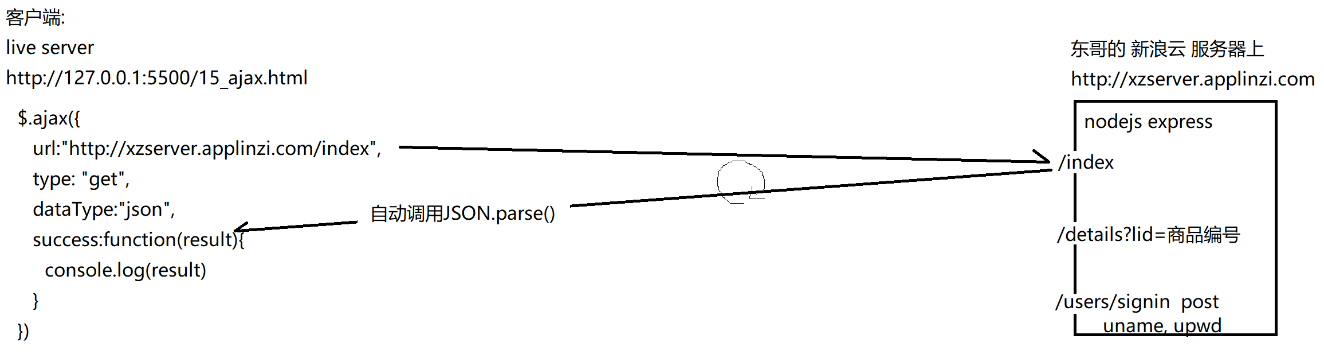
**})**

3. 强调:

(1). $.ajax()使用的是类似ES6参数结构的语法，所以，所有实参值必须凡在一个对象结构中。且对象结构中**:前的属性名绝对不能改名**！因为必须和人家函数中规定的形参变量名保持一致才行——复习ES6的参数解构

(2). 因为**$.ajax()是异步请求**，所以，如果**希望在ajax请求响应成功结束后才能执行的所有js代码，必须都放在success:function(){}内调用或执行！**不能放在success外部调用或执行！

4. 示例: 使用$.ajax()向学子商城服务器端发送请求，获取响应结果



15\_ajax.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>    <script>      //想从东哥新浪云服务器中请求学子商城首页的6个商品对象组成的数组。      //已知: 服务端接口地址:http://xzserver.applinzi.com/index      //类型: get      //不需要参数:      $.ajax({        //url:"http://xzserver.applinzi.com/index",        url:"http://www.codeboy.com:9999/data/product/index.php",        type: "get",        dataType:"json", //说明服务器端返回的是json字符串，需要自动调用JSON.parse()转为js对象或数组        success:function( result ){ //只在请求响应成功结束后才自动执行！          //result就是服务器端返回的响应结果，且已经被JSON.parse()转为了js的数组或对象，不再需要手工调用JSON.parse()了！          console.log(result);        }      })      //想从东哥新浪云服务器/details接口获得5号商品的详细信息      //类型: get      //参数: lid=商品编号      $.ajax({        // url:"http://xzserver.applinzi.com/details",        url:"http://www.codeboy.com:9999/data/product/details.php",        type: "get",        data:{ lid:5 },        dataType:"json" ,        success:function( result ){          console.log(result);        }      })      //想用东哥新浪云服务器的登陆接口验证用户名和密码      //类型: post      //参数: uname=用户名  upwd=密码      $.ajax({        // url:"http://xzserver.applinzi.com/users/signin",        url:"http://www.codeboy.com:9999/data/user/login.php",        type: "post",        data:{ uname:"dingding", upwd:"123456" },        dataType:"json" ,        success:function( result ){          console.log(result);        }      })    </script>  </body>  </html>  运行结果:  {carouselItems: Array(4), recommendedItems: Array(6), newArrivalItems: Array(6), topSaleItems: Array(6)}  carouselItems: (4) [{…}, {…}, {…}, {…}]  newArrivalItems: (6) [{…}, {…}, {…}, {…}, {…}, {…}]  recommendedItems: (6) [{…}, {…}, {…}, {…}, {…}, {…}]  topSaleItems: (6) [{…}, {…}, {…}, {…}, {…}, {…}]  \_\_proto\_\_: Object  {details: {…}, family: {…}}  details: {lid: "5", family\_id: "2", title: "小米(MI)Air 13.3英寸全金属超轻薄笔记本(i5-6200U 8G 256G PCIE固态 940MX独显 FHD WIN10)银", subtitle: "【i5 独立显卡】全高清窄边框 8G内存 256G固态硬盘 支持SSD硬盘扩容 薄至14.8mm 轻至1.28kg！AKG扬声器！", price: "4999.00", …}  family: {fid: "2", fname: "小米Air", laptopList: Array(4)}  \_\_proto\_\_: Object  {code: 200, msg: "login succ"}  code: 200  msg: "login succ"  \_\_proto\_\_: Object |

四. \*\*\*跨域

对应小程序视频: 小程序->在线->JQUERY->day04 16 jquery跨域cors ...

1. 什么是: 一个网站下的网页去另一个域名下的网站请求资源(文件、数据)

2. 比如:

<img src="其它网站的图片地址">

<a href="其它网站的网页">

<link rel="stylesheet" href="其它网站的css">

<script src="其它网站的js">

在淘宝或支付宝中，可以查看几乎所有快递公司的快递单进度

3. 包括:

(1). 域名不同: 从 http://www.a.com 下的网页 请求 http://www.b.com下的资源

(2). 子级域名不同: 从 http://oa.tedu.com 下的网页 请求 http://hr.tedu.com下的资源

(3). 端口不同: 从 http://localhost:5050 下的网页 请求 http://localhost:3000 下的接口

(4). 协议不同: 从 http://12306.cn 下的网页 请求 https://12306.cn 下的资源

(5). 即使同一台机器，域名与IP之间互访:

从 http://localhost 下的网页 请求 http://127.0.0.1下的资源

4. 浏览器**同源策略**（CORS）:

(1). 什么是: 浏览器中有一个安全策略，要求一个网页只能用ajax请求来自自己网站的数据。不允许ajax请求其它网站的数据。

(2). 何时: 如果用ajax从一个网站的网页，向另一个域名下的网站发送请求，就会触发浏览器的同源策略:

(3). 结果: 报错: Access to XMLHttpRequest at 'http://localhost:3000/' from origin 'http://127.0.0.1:5500' has been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present on the requested resource. (这个错误可能因为浏览器版本不同略有不同，但是大致相同！)

翻译: 从源头'http://127.0.0.1:5500' 向 'http://localhost:3000/' 发送的访问请求 被CORS 策略阻止了。因为在响应头中没有设置'Access-Control-Allow-Origin'属性！

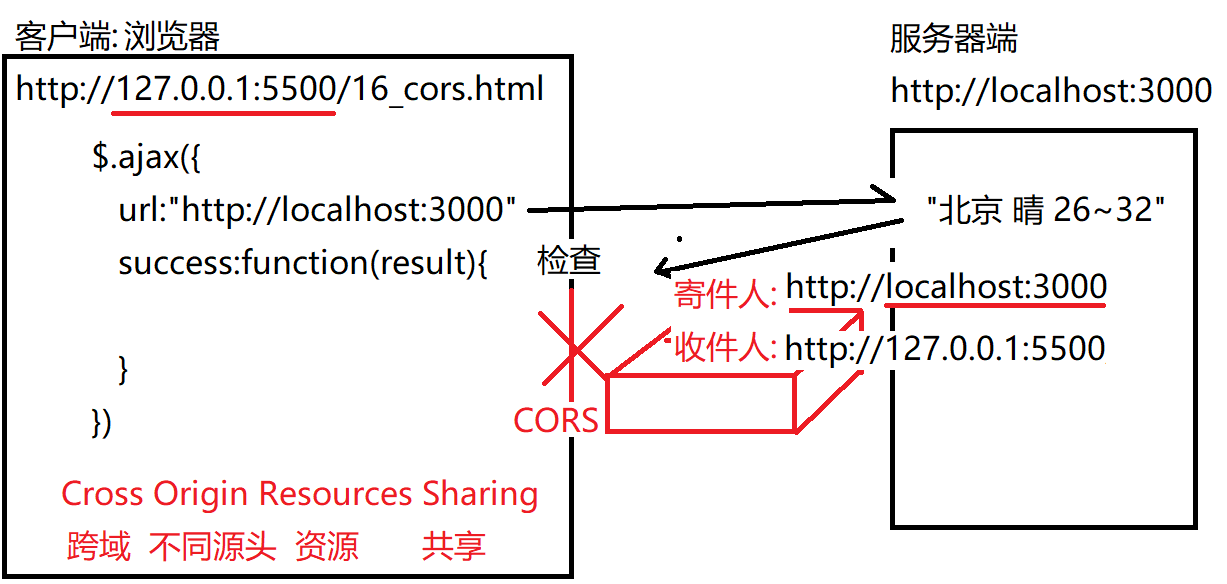
(4). 原理:

a. $.ajax可以发送ajax请求到任何服务器端

b. 但是，服务器端返回结果后，客户端浏览器会检查返回结果的来源地址

c. 默认，服务器端返回结果会原原本本的记录真实服务器所在域名地址

d. 浏览器如果发现服务器端返回的结果上记录的数据来源地址与当前网页所在的域名地址不同！浏览器就会禁止ajax使用该返回结果！



5. 解决跨域问题: 3种方式:

**(1). CORS方式: 欺骗方式**

a. 让服务器端在返回响应结果之前，在**响应头**中添加一个特殊属性: 'Access-Control-Allow-Origin'，属性值为服务器端要伪装的客户端地址。

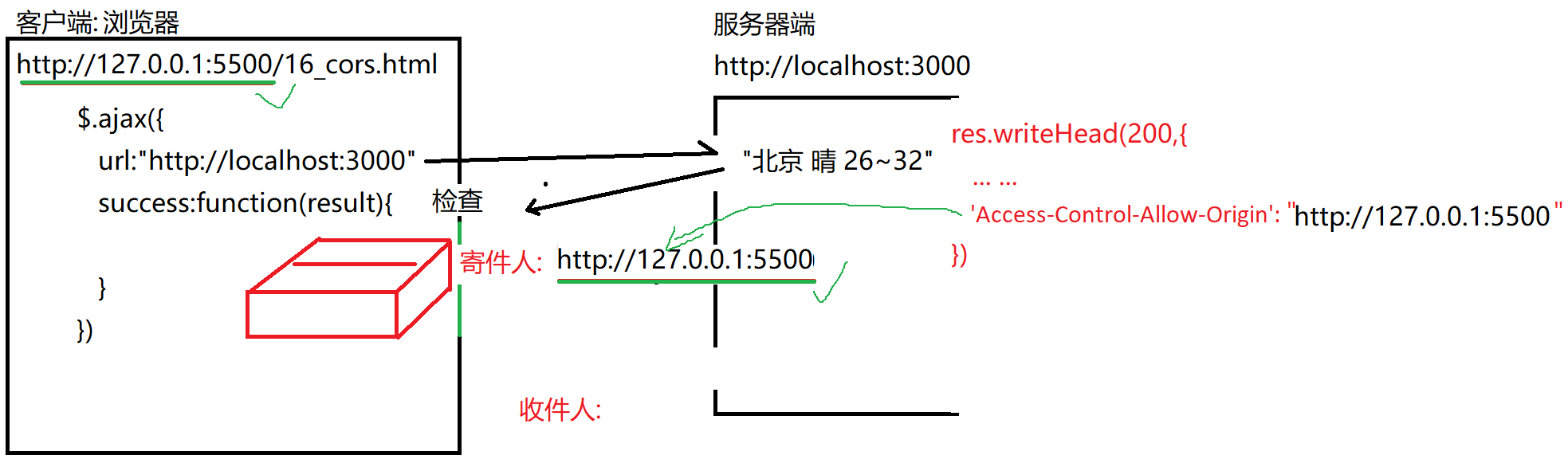
**res.writeHead(200,{**

**... ...**

**'Access-Control-Allow-Origin': "客户端地址"**

**寄件人**

**})**



b. 结果: 服务器端返回的响应结果中的寄件人地址被伪装成了和客户端网页所在地址一样！则就能顺利通过浏览器CORS策略的检查！

(2). JSONP：基础比较好的同学看小程序扩展视频

(3). 服务器端中继：纯服务器端，我们不需要了解。

6. 示例: 使用服务器端cors方式解决跨域问题:

16\_server.js

|  |
| --- |
| //不用写！！！  //也不要求会写！！！！  //只要会运行就行！！！  //声明常量http，引入nodejs标准模块http模块，将引入的对象保存在http常量中  const http=require("http");  //创建一个服务器端的程序server，监听一个指定的端口。  var server=http.createServer(    //当这个端口收到从客户端发开的请求时，自动调用回调函数    //回调函数的req参数，保存了本次请求相关的信息    //回调函数的res参数，保存了本次响应相关的信息    function(req,res){      //只要服务端程序收到任何请求      //都返回一个字符串响应结果:"北京 晴 26~32"      //因为用的不是express框架，所以，不能用res.send()只能写res.write() res.end()      var weather="北京 晴 26~32";      //修改响应头中的配置信息      //告诉客户端浏览器      //            响应成功      res.writeHead(200,{      //服务器返回的内容类型为普通文本，且字符编码是utf-8格式        "Content-Type":"text/plain;charset=utf-8",      //结果: 浏览器就会自动用utf-8格式解析服务器端返回的内容      //将服务器端返回结果中的寄件人地址伪装成和客户端网页地址相同        "Access-Control-Allow-Origin":"http://127.0.0.1:5500"      //结果: 服务器端返回的结果中寄件人被伪装成http://127.0.0.1:5500，和客户端所在地址完全相同，就能顺利通过浏览器的CORS策略检查！      })      //将北京的天气字符串，先写入响应对象中暂存。      res.write(weather);      //让响应对象res，将保存的北京天气字符串返回给客户端      res.end();    }  )  //让服务器端程序server监听3000端口  server.listen(3000);  //如何启动服务端程序:  //右键单击16\_server.js  //选择在xx终端中打开  //等弹出窗口中出现C:\xxx\xxxx > 才能继续输入  //输入node 16\_server.js  //手工打开浏览器，地址栏输入: http://localhost:3000  //结果: 看到服务器端程序返回的北京天气字符串  //无论以上服务器端程序作了任何修改，保存后，都要重启服务端:  //  先Ctrl+C 退出旧的服务器端程序  //  再运行node 16\_server.js  //  重新刷新浏览器中的页面 |

16\_cors.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>    <script>      $.ajax({        url:"http://localhost:3000",        type:"get",        success:function(result){          console.log(result);        }      })    </script>  </body>  </html>  运行结果:  北京 晴 26~32 |

总结: jq中$共有4种用法:

1. $("选择器") 查找DOM元素，并保存进新创建的jquery对象中

2. $(DOM元素对象) 无需查找，直接将已经获得的DOM元素，保存进新创建的jquery对象中

3. $(`HTML片段`) 新创建一个HTML元素，保存进新创建的jquery对象中

4. $(function(){ ... }) 绑定DOMContentLoaded事件！

总结: jQuery API三大特点:

1. 自带遍历:

2. 一个函数两用:

3. 返回当前正在操作的.前的jQuery对象。——链式操作！

总结：

9. 事件绑定:

a. 标准写法: $元素.on("事件名", 事件处理函数)

b. 简写: $元素.事件名(事件处理函数)

c. 事件委托:

$父元素.on("事件名","选择器",function(){

...this指向e.target...

})

d. 页面加载后自动执行:

1). 先$(document).ready(function(){})

简写: $(function(){ ... })

2). 后$(window).load(function(){ ... })

e. 鼠标事件:

mouseenter代替mouseover

mouseleave 代替mouseout

简写: .hover(处理函数1,处理函数2)

等于: .mouseenter(处理函数1)

.mouseleave(处理函数2)

f. 模拟触发:

1). 标准: $元素.trigger("事件名")

2). 如果属于常用事件列表，可简写为:

$元素.事件名()

10. 动画:

a. 简单动画: 3种固定效果

1). 显示隐藏:

.show() .hide() . toggle()

2). 淡入淡出:

.fadeIn() .fadeOut() .fadeToggle()

3). 上滑下滑:

.slideUp() .slideDown() .slideToggle()

b. 万能动画:

$元素.animate({

css属性: 目标值

},动画持续时间,function(){

动画播放结束自动执行

})

c. 排队和并发:

1). 放在一个animate()中的多个css属性并发变化

2). 放在一个元素的多个animate()中的多个css属性排队变化

d. 停止动画:

1). 只停止当前一个动画

$元素.stop()

2). 停止队列中所有动画

$元素.stop(true)

e. 选择器匹配正在播放动画的元素: :animated

11. 类数组对象操作:

1). 遍历查找结果中每个DOM元素对象:

$查找结果.each(function(i, domElem){ ... })

2). 查找一个DOM元素在整个查找结果中的下标位置

var i=$查找结果.index(要找的DOM元素)

12. 添加自定义函数:

a. 添加jQuery.fn.自定义函数=function(){ ... }

b. 调用: $jq对象.自定义函数()  
13. 封装自定义插件:

a. 封装:

1). 将css提取到独立的插件.css文件中

2). 在独立js文件中为jquery原型对象添加自定义插件函数

jQuery.fn.自定义插件函数=function(){

//自动做2件事:

//1. 为插件所在元素及其子元素自动添加class  
 //2. 将原网页中插件所需的事件绑定代码剪切到插件函数中

}

b. 使用自定义插件:

1). 引入jquery.js，插件.js，插件.css

2). 按插件要求编写HTML内容，不用加任何class

3). 在自定义js中，查找插件所在元素，调用自定义插件函数

14. 发送ajax请求:

$.ajax({

url:"服务器端接口地址",

type:"get或post",

data:{ 参数名: 参数值, ... },

dataType:"json",

success:function(result){

//result就是服务器端返回的结果

//不用自己调JSON.parse()

}

})

15. 跨域:

a. 同源策略(CORS): 浏览器只允许当前网页中的ajax请求和使用自己网站的资源，禁止ajax请求和使用其他网站的资源。

b. 解决: cors方式：只靠服务器端就能完成:

res.writeHead(200,{  
 "Access-Control-Allow-Origin": "http://客户端网页所在网址"  
})

c. 结果: 服务器端将响应结果中的寄件人地址伪装成和客户端网页所在地址一致，就可骗过浏览器cors策略的检查，让ajax顺利使用其他网站服务器端返回的结果数据了。

今日对应小程序视频:

小程序->在线->JQUERY->day04: 14 添加jQUery自定义函数...

15 jQuery发送ajax请求$.ajax

16 jquery跨域cors ...

作业:

1. 复习今日小程序问题清单: 小程序->首页->JQUERY->day04

2.（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