**跨域问题：**

baidu的通行证处理都是在二级域名passport.baidu.com中处理的，但是baidu很多地方登录都好像是用ajax处理的，他是怎么做的呢？研究了一下，发现一个小技巧。

在http://zhidao.baidu.com/ 未登录用户回答问题时会用iframe调用http://zhidao.baidu.com/userlogin.html，userlogin.html 有下面的 javascript：

|  |  |  |
| --- | --- | --- |
| 01 | <SCRIPT LANGUAGE="JavaScript"> | |
| 02 | document.domain="baidu.com"; |

|  |  |
| --- | --- |
| 03 | <!-- |
| 04 | function G(id){if(typeof(id)=="string"){return document.getElementById(id);}return id;} | |

|  |  |
| --- | --- |
| 05 | function showInfo(obj){ |
| 06 | if(obj.checked == true){ | |

|  |  |  |
| --- | --- | --- |
| 07 | G("memInfo").style.display="block"; | |
| 08 | }else{ |

|  |  |  |
| --- | --- | --- |
| 09 | G("memInfo").style.display="none"; | |
| 10 | } |

|  |  |
| --- | --- |
| 11 | } |
| 12 | function request(id,url){ | |

|  |  |
| --- | --- |
| 13 | oScript = document.getElementById(id); |
| 14 | var head = document.getElementsByTagName("head").item(0); | |

|  |  |
| --- | --- |
| 15 | if (oScript) { |
| 16 | head.removeChild(oScript); | |

|  |  |
| --- | --- |
| 17 | } |
| 18 | oScript = document.createElement("script"); | |

|  |  |  |
| --- | --- | --- |
| 19 | oScript.setAttribute("src", url); | |
| 20 | oScript.setAttribute("id",id); |

|  |  |  |
| --- | --- | --- |
| 21 | oScript.setAttribute("type","text/javascript"); | |
| 22 | oScript.setAttribute("language","javascript"); |

|  |  |  |
| --- | --- | --- |
| 23 | head.appendChild(oScript); | |
| 24 | return oScript; |

|  |  |
| --- | --- |
| 25 | } |
| 26 | var loginTimer=null; | |

|  |  |  |
| --- | --- | --- |
| 27 | var loginState=-1; | |
| 28 | var tryTime=0; |

|  |  |  |
| --- | --- | --- |
| 29 | function PSP\_ik(isOk){ | |
| 30 | if(isOk==0){ |

|  |  |  |
| --- | --- | --- |
| 31 | G("errorInfo").style.display="none"; | |
| 32 | loginState=1; |

|  |  |
| --- | --- |
| 33 | if(parent.loginSuccess){ |
| 34 | parent.Pop.hide(); | |

|  |  |  |
| --- | --- | --- |
| 35 | parent.loginSuccess(); | |
| 36 | } |

|  |  |
| --- | --- |
| 37 | } |
| 38 | else | |

|  |  |
| --- | --- |
| 39 | { |
| 40 | loginFalse(); | |

|  |  |  |
| --- | --- | --- |
| 41 | } | |
| 42 | } |

|  |  |
| --- | --- |
| 43 |  |
| 44 | function loginFalse(){ | |

|  |  |
| --- | --- |
| 45 | loginState=0; |
| 46 | var err=G("errorInfo"); | |

|  |  |  |
| --- | --- | --- |
| 47 | err.innerHTML="用户名或密码错误,请重新登录"; | |
| 48 | err.style.display="block"; |

|  |  |  |
| --- | --- | --- |
| 49 | G("username").focus(); | |
| 50 | tryTime++; |

|  |  |
| --- | --- |
| 51 | if(tryTime>1){ |
| 52 | onLoginFailed(); | |

|  |  |  |
| --- | --- | --- |
| 53 | } | |
| 54 | } |

|  |  |
| --- | --- |
| 55 | function onLoginFailed(){ |
| 56 | if(parent.onLoginFailed){ | |

|  |  |
| --- | --- |
| 57 | parent.Pop.hide(); |
| 58 | parent.loginFailed(); | |

|  |  |  |
| --- | --- | --- |
| 59 | | }else{ |
| 60 | document.login.u.value=escape("<http://zhidao.baidu.com/q>"+parent.location.search); | | |

|  |  |  |
| --- | --- | --- |
| 61 | doucment.login.submit(); | |
| 62 | } |

|  |  |  |
| --- | --- | --- |
| 63 |  | |
| 64 | } |

|  |  |
| --- | --- |
| 65 | function loginTimeout(){ |
| 66 | if(loginState==-1){ | |

|  |  |
| --- | --- |
| 67 | var err=G("errorInfo"); |
| 68 | err.innerHTML="操作超时,请重新登录"; | |

|  |  |  |
| --- | --- | --- |
| 69 | err.style.display="block"; | |
| 70 | G("username").focus(); |

|  |  |  |
| --- | --- | --- |
| 71 | } | |
| 72 | } |

|  |  |
| --- | --- |
| 73 | function userLogin(){ |
| 74 | var username=G('username').value; | |

|  |  |
| --- | --- |
| 75 | var password=G('password').value; |
| 76 | var memPassport=G('memPassport').checked?"on":"off"; | |

|  |  |  |
| --- | --- | --- |
| 77 | | if(username.length<=0||password.length<=0){G("username").focus();return false;} |
| 78 | var url = '<https://passport.baidu.com/?logt&tpl=ik&t=0&keyname=ik&mem_pass=>'+memPassport+'&username='+username + '&loginpass=' +escape(password)+ '&s=' + (new Date()).getTime(); | |

|  |  |
| --- | --- |
| 79 | loginState=-1; |
| 80 | var login=request("loginScript",url); | |

|  |  |  |
| --- | --- | --- |
| 81 | loginTimer = setTimeout(loginTimeout, 5000); | |
| 82 |  |

|  |  |
| --- | --- |
| 83 | } |
| 84 | window.onload=function(){ | |

|  |  |
| --- | --- |
| 85 | document.loginForm.username.focus(); |
| 86 | document.getElementById("username").focus(); | |

|  |  |
| --- | --- |
| 87 | } |
| 88 | //--> | |

|  |  |
| --- | --- |
| 89 | </SCRIPT> |

我们可以看到request方法处理异步请求使用动态往head中添加script而不是用xmlhttp发送get请求。妙就妙在这。我们知道调用javascript是没有域的限制的。当加载完成时一样会执行。

当然请求参数只能通过拼url的方式了。url通过服务器处理后直接输出loginFalse()或者PSP\_ik()；非常优雅的解决了跨域的问题。

这让我们想到了用iframe当ajax上传文件一样异曲同工。如果不需要服务器反馈,google的点击计数用new img().src=...;

当然baidu这段脚本中还有一些小的技巧也值得我们学习。

**JavaScript**出于安全方面的考虑，不允许跨域调用其他页面的对象。跨域就是因为JavaScript同 源策略的限制。在客户端编程语言中，如javascript和ActionScript，同源策略是一个很重要的安全理念，它在保证数据的安全性方面有着 重要的意义。同源策略规定跨域之间的脚本是隔离的，一个域的脚本不能访问和操作另外一个域的绝大部分属性和方法。下面让我们共同学习一下跨越共享的方法。

**同源策略**

在客户端编程语言中，如javascript和ActionScript，同源策略是一个很重要的安全理念，它在保证数据的安全性方面有着重要的意 义。同源策略规定跨域之间的脚本是隔离的，一个域的脚本不能访问和操作另外一个域的绝大部分属性和方法。那么什么叫相同域，什么叫不同的域呢？当两个域具 有相同的协议(如http),相同的端口(如80)，相同的host（如www.example.org)，那么我们就可以认为它们是相同的域。比如 http://www.example.org/index.html和http://www.example.org/sub/index.html是 同域，而http://www.example.org,https://www.example.org,http: //www.example.org:8080,http://sub.example.org中的任何两个都将构成跨域。同源策略还应该对一些特殊情况 做处理，比如限制file协议下脚本的访问权限。本地的HTML文件在浏览器中是通过file协议打开的，如果脚本能通过file协议访问到硬盘上其它任 意文件，就会出现安全隐患，目前IE8还有这样的隐患。

受到同源策略的影响，跨域资源共享就会受到制约。但是随着人们的实践和浏览器的进步，目前在跨域请求的技巧上，有很多宝贵经验的沉淀和积累。这里我 把跨域资源共享分成两种，一种是单向的数据请求，还有一种是双向的消息通信。

|  |  |  |
| --- | --- | --- |
| URL | 说明 | 是否允许通信 |
| http://www.kuqin.com/lab/a.js http://www.kuqin.com/script/b.js | 同一域名下不同文件夹 | 允许 |
| http://www.kuqin.com/a.js http://www.kuqin.com/b.js | 同一域名下 | 允许 |
| http://www.kuqin.com:8000/a.js http://www.kuqin.com/b.js | 同一域名，不同端口 | 不允许 |
| http://www.kuqin.com/a.js https://www.kuqin.com/b.js | 同一域名，不同协议 | 不允许 |
| http://www.kuqin.com/a.js http://70.32.92.74/b.js | 域名和域名对应ip | 不允许 |
| http://www.kuqin.com/a.js http://script.kuqin.com/b.js | 主域相同，子域不同 | 不允许 |
| http://www.ithao123.com/a.js http://www.kuqin.com/b.js | 不同域名 | 不允许 |

特别注意两点：  
  
第一，如果是协议和端口造成的跨域问题“前台”是无能为力的，

第二：在跨域问题上，域仅仅是通过URL的首部来识别而不会去尝试判断相同的ip地址对应着两个域或两个域是否在同一个ip上。

**1、document.domain+iframe的设置**  
对于主域相同而子域不同的例子，可以通过设置 document.domain的办法来解决。具体的做法是可以在http://www.kuqin.com/a.html和http: //script.kuqin.com/b.html两个文件中分别加上document.domain = ‘kuqin.com’；然后通过a.html文件中创建一个iframe,去控制iframe的contentDocument,这样两个js文件之间 就可以“交互”了。当然这种办法只能解决主域相同而二级域名不同的情况，如果你异想天开的把script.kuqin.com的domian设为 alibaba.com那显然是会报错地！代码如下：  
www.kuqin.com上的a.html

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | document.domain = "kuqin.com";  **var** ifr = document.createElement("iframe");  ifr.src = "http://script.kuqin.com/b.html";  ifr.style.display = "none";  document.body.appendChild(ifr);  ifr.onload = **function**(){  **var** x = ifr.contentDocument;  alert(x.getElementsByTagName("h1")[0].childNodes[0].nodeValue);  } |

script.kuqin.com上的b.html

|  |  |
| --- | --- |
| 1 | document.domain = "kuqin.com"; |

**2、动态创建script**  
虽然浏览器默认禁止了跨域访问，但并不禁止在页面中引用其他域的JS文件，并可以自由执行引入的JS文件中的function，根据这一点，可以方便地通过创建script节点的方法来实现完全跨域的通信。具体的做法可以参考yui的 [Get Utility](http://developer.yahoo.com/yui/get/)

这里判断script节点加载完毕还是蛮有意思的：ie只能通过script的readystatechange属性,Safari 3.x以上支持的是script的load事件，而firefox和oprea则要通过onload来解决。另外这种办法只能传递js类型的数据，不是很 方便。以下是部分判断script加载完毕的方法。

**3、利用iframe和location.hash**  
这个办法比较绕，但是可以解决完全跨域情况下的脚步 置换问题。原理是利用location.hash来进行传值。在url： http://kuqin.com#helloword中的‘#helloworld’就是location.hash,改变hash并不会导致页面刷 新，所以可以利用hash值来进行数据传递，当然数据容量是有限的。假设域名kuqin.com下的文件cs1.html要和ithao123.com域 名下的cs2.html传递信息，cs1.html首先创建自动创建一个隐藏的iframe，iframe的src指向ithao123.com域名下的 cs2.html页面，这时的hash值可以做参数传递用。cs2.html响应请求后再将通过修改cs1.html的hash值来传递数据。（因为ie 不允许修改parent.location.hash的值，所以要借助于kuqin.com域名下的一个代理iframe）。同时在cs1.html上加 一个定时器，隔一段时间来判断location.hash的值有没有变化，一点有变化则获取获取hash值。代码如下：  
先是kuqin.com下的文件cs1.html文件：

kuqin.com下的域名cs3.html

|  |  |
| --- | --- |
| 1  2 | *//因为parent.parent和自身属于同一个域，所以ie下可以改变其location.hash的值*  parent.parent.location.hash = self.location.hash.substring(1); |

实例请点击  [hash实现完全跨域](http://f2e.me/test/cscript/cs1.html)  
当然这样做也存在很多缺点，诸如数据直接暴露在了url中，数据容量和类型都有限等……

**4、利用flash**  
这是从YUI3的IO组件中看到的办法,具体可见：<http://developer.yahoo.com/yui/3/io/>  
flash这个方案不是很明白，各位自己慢慢琢磨了，呵呵。你可以看在Adobe Developer Connection看到更多的跨域代理文件规范：  
[ross-Domain Policy File Specifications.](http://www.adobe.com/devnet/articles/crossdomain_policy_file_spec.html)  
[HTTP Headers Blacklist.](http://kb.adobe.com/selfservice/viewContent.do?externalId=kb403030)

域(Domain)是Windows网络中独立运行的单位，域之间相互访问则需要建立信任关系(即 Trust Relation)。信任关系是连接在域与域之间的桥梁。当一个域与其他域建立了信任关系后，2个域之间不但可以按需要相互进行管理，还可以跨网分配文件 和打印机等设备资源，使不同的域之间实现网络资源的共享与管理。 有一种简明的说法来解释广域跨域：跨域访问，简单来说就是 A 网站的 javascript 代码试图访问 B 网站，包括提交内容和获取内容。由于安全原因，跨域访问是被各大浏览器所默认禁止的。

在广域网环境中，由于浏览器的安全限制，网络连接的跨域访问时不被允许的，XmlHttpRequest也不例外。但有时候跨域访问资源是必需的。

同源策略阻止从一个域上加载的脚本获取或操作另一个域上的文档属性。也就是说，受到请求的 URL 的域必须与当前 Web 页面的域相同。这意味着浏览器隔离来自不同源的内容，以防止它们之间的操作。同源策略不阻止将动态脚本元素插入文档中。

参考理论一：在浏览器中不能直接来跨域访问，而在服务器端没有跨域安全限制。

这样的话，可以在服务端完成跨域访问，而在客户端来取得结果就可以了。

参考理论二：同源策略不阻止动态脚本元素插入，脚本访问可以跨域。

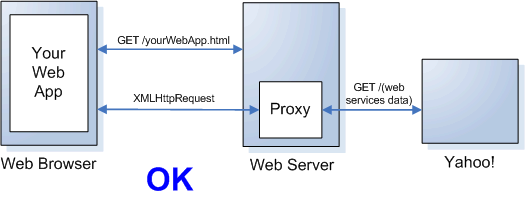
**用服务器端的XmlHttpRequest代理实现跨域访问**

我们不能在浏览器端直接使用AJAX来跨域访问资源，但是在服务器端是没有这种跨域安全限制的。所以， 我们只需要让服务器端帮我们完成“跨域访问”的工作，然后在浏览器端用AJAX获取服务器端“跨域访问”的结果就可以了。这就是所谓的在服务器端创建一个 XmlHttpRequest代理，通过这个代理来访问其他域名下的资源。这里引用Yahoo! JavaScript Developer Center上的几张图来进一步说明这个方案：

使用XmlHttpRequest访问同一域名下的资源：

使用XmlHttpRequest跨域访问资源：

用服务器端的XmlHttpRequest代理来跨域访问资源：



编写服务器端XmlHttpRequest代理的具体过程就不赘述了，无非是创建一个自定义的HTTP请求。

**其他实现跨域访问的方案**

除了上面两种方案之外，还有其他一下方法同样可以实现跨域访问，例如：

* apache server的mod\_rewrite模块或者mod\_proxy模块可以帮你自动实现服务器端的XmlHttpRequest代理，你可以像调用当前域名下的资源一样的调用其他域名下的资源。
* 据说动态生成iframe也可以实现跨域调用，没有仔细研究过。

**Javascript跨域和Ajax跨域解决方案**

Ajax跨域和JS的跨域通信(Cross The Site)的几种解决方案

最近做的一个项目中需要ajax跨域取得数据,如果是在本域中确实没有问题，但是放到二级域和其他域下浏览器直接就弹出提示框:“该页正在访问其控制范围之外的数据，这有些危险，是否继续"。

什么引起了ajax跨域不能的问题？ajax本身实际上是通过XMLHttpRequest对象来进行数据的交互，而浏览器出于安全考虑，不允许js代码进行跨域操作，所以会警告。

有什么完美的解决方案么？没有。解决方案有不少，但是只能是根据自己的实际情况来选择。

具体情况有：

* 本域和子域的相互访问: www.aa.com和book.aa.com
* 本域和其他域的相互访问: www.aa.com和www.bb.com 用 iframe
* 本域和其他域的相互访问: www.aa.com和www.bb.com 用 XMLHttpRequest访问代理
* 本域和其他域的相互访问: www.aa.com和www.bb.com 用 JS创建动态脚本

解决方法：

* 如果想做到数据的交互，那么www.aa.com和book.aa.com必须由你来开发才可以。可以将book.aa.com用 iframe添加到 www.aa.com的某个页面下，在www.aa.com和iframe里面都加上document.domain = "aa.com"，这样就可以统一域了，可以实现跨域访问。就和平时同一个域中镶嵌iframe一样，直接调用里面的JS就可以了。
* 当两个域不同时，如果想相互调用，那么同样需要两个域都是由你来开发才可以。用iframe可以实现数据的互相调用。解决方案就是用 window.location对象的hash属性。hash属性就是http://domian/web/a.htm#dshakjdhsjka 里面的#dshakjdhsjka。利用JS改变hash值网页不会刷新，可以这样实现通过JS访问hash值来做到通信。不过除了IE之外其他大部分浏 览器只要改变hash就会记录历史，你在前进和后退时就需要处理，非常麻烦。不过再做简单的处理时还是可以用的。大体的过程是页面a和页面b在不同域 下,b通过iframe添加到a里，a通过JS修改iframe的hash值，b里面做一个监听（因为JS只能修改hash，数据是否改变只能由b自己来 判断），检测到b的hash值被修改了，得到修改的值，经过处理返回a需要的值，再来修改a的hash值（这个地方要注意，如果a 本身是那种查询页面的话比如http://domian/web/a.aspx?id=3,在b中直接parent.window.location是无 法取得数据的，同样报没有权限的错误，需要a把这个传过来，所以也比较麻烦），同样a里面也要做监听，如果hash变化的话就取得返回的数据，再做相应的 处理。
* 这种情形是最经常遇到的，也是用的最多的了。就是www.aa.com和www.bb.com你只能修改一个，也就是另外一个是别人的， 人家告诉你你要取得数据就访问某某连接参数是什么样子的，最后返回数据是什么格式的。而你需要做的就是在你的域下新建一个网页，让服务器去别人的网站上取 得数据，再返回给你。domain1下的a向同域下的GetData.aspx请求数据，GetData.aspx向domain2下的 ResponseData.aspx 发送请求，ResponseData.aspx 返回数据给 GetData.aspx，GetData.aspx 再返回给 a，这样就完成了一次数据请求。GetData.aspx在其中充当了代理的作用。
* 这个和上个的区别就是请求是使用<script>标签来请求的，这个要求也是两个域都是由你来开发才行。原理就是JS文件注 入，在本域内的a 内生成一个JS标签，它的SRC指向请求的另外一个域的某个页面b，b返回数据即可，可以直接返回JS的代码。因为script的src属性是可以跨域 的。

总结：

* 第一种情况：域和子域的问题，可以完全解决交互。
* 第二种情况：跨域，实现过程非常麻烦，需要两个域开发者都能控制，适用于简单交互。
* 第三种情况：跨域，开发者只控制一个域即可，实现过程需要增加代理取得数据，是常用的方式。
* 第四种情况：跨域，两个域开发者都需要控制，返回一段js代码。

**google下拉框**

JSP页面

Html代码  [收藏代码](javascript:void())

1. <script type="text/javascript" src="../javascript/jquery.js"></script>
2. <script type="text/javascript" src="../javascript/auto.js"></script>
3. <body>
4. Google Suggest <input id="word" type="text">
5. <input type="button" value="Click"><br/>
6. <div id="auto"></div>
7. </body>
8. </html>

JS文件

1. //全局变量，高亮的信息-1表示没有选中
2. var highlightindex = -1;
3. //延时处理
4. var timeout;
5. $(document).ready(function(){
6. var wordInput = $("#word");
7. //获得JQUERY封装的一个对象
8. var wordInputOffset = wordInput.offset();
9. //设置DIV为隐藏，并设置边框属性
10. $("#auto").hide().css("border","1px black solid")
11. .css("position","absolute")
12. //input离上面的长度+input的高（宽）
13. .css("top",wordInputOffset.top+wordInput.height()+5+"px")
14. //input高左面的的长度
15. .css("left",wordInputOffset.left+"px")
16. //input的宽度
17. .width(wordInput.width()+6+"px");
18. //添加事件
19. $("#word").keyup(function(event){
20. var myEvent = event || window.event;
21. var myKeyCode = myEvent.keyCode;
22. //给页面的DIV做一个缓存
23. var div = $("#auto");
24. //获得文件框里面的值
25. if(myKeyCode>64 && myKeyCode<91 || myKeyCode==8 || myKeyCode==46){
26. var wordText = $("#word").val();
27. if(wordText!=""){
28. //延时操作
29. clearTimeout(timeout);
30. timeout = setTimeout(function(){
31. //和服务器端交互，第四个参数设置返回是XML数据
32. $.post("../servlet/AutoComplete",{word:wordText},function(data){
33. //万能的$获得XML数据
34. var xmlobj = $(data);
35. //找到其中的WORD标签
36. var wordNodes = xmlobj.find("word");
37. div.html("");
38. //循环输出wordNodes
39. wordNodes.each(function(i){
40. //$(this)表示当前的WORD字段
41. var wordNode = $(this);
42. //创建 DIV并,设值，再添加到页面的DIV上.(还可以属性attr("value",text); )
43. var newDiv = $("<div>").attr("id",i);
44. newDiv.html(wordNode.text()).appendTo(div);
45. newDiv.mouseover(function(){
46. if(highlightindex!=-1){
47. $("#auto").children("div").eq(highlightindex).css("background-color","white");
48. }
49. highlightindex = $(this).attr("id");
50. $(this).css("background-color","blue");
51. });
52. newDiv.mouseout(function(){
53. $(this).css("background-color","white");
54. });
55. newDiv.click(function(){
56. $("#auto").hide();
57. var textClick = $("#auto").children("div").eq(highlightindex).text();
58. highlightindex = -1;
59. $("#word").val(textClick);
60. });
61. });
62. //如果返回的wordNodes长度大于0就show
63. if(wordNodes.length > 0){
64. div.show();
65. }else{
66. div.hide();
67. highlightindex = -1;
68. }
69. },"xml");
70. },1000);
72. }else{
73. //如果文本信息为空就隐藏
74. div.hide();
75. highlightindex = -1;
76. }
78. //键盘向上向下键
79. }else if(myKeyCode == 38 || myKeyCode ==40){
80. //向上
81. if(myKeyCode == 38){
82. //获得ID为auto里面的DIV对象
83. var autoNodes = $("#auto").children("div");
84. //如果没有被选中
85. if(highlightindex != -1){
86. //设置CSS
87. autoNodes.eq(highlightindex).css("background-color","white");
88. highlightindex--;
89. }else{
90. highlightindex = autoNodes.length - 1;
91. }
92. autoNodes.eq(highlightindex).css("background-color","blue");
93. }
94. //向下
95. if(myKeyCode == 40){
96. var autoNodes = $("#auto").children("div");
97. if(highlightindex != -1){
98. autoNodes.eq(highlightindex).css("background-color","white");
99. highlightindex++;
100. }else{
101. highlightindex++;
102. }
103. if(highlightindex == autoNodes.length){
104. highlightindex = 0;
105. }
106. autoNodes.eq(highlightindex).css("background-color","blue");
107. }
109. //回车键
110. }else if(myKeyCode ==13){
111. if(highlightindex!=-1){
112. //获得选中的那个的文本值
113. var textInputText = $("#auto").children("div").eq(highlightindex).text(
114. highlightindex = -1;
115. $("#word").val(textInputText);
116. $("#auto").hide();
117. }else{
118. alert("提交服务器");
119. $("#auto").hide();
120. //失去焦点
121. $("#word").get(0).blur();
122. }
123. }
124. });
125. //获得一个input对象，并且他的type属性为button
126. $("input[type='button']").click(function(){
127. alert("提交服务器");