跨域

什么是跨域

域名地址的组成：

http:// www . google : 8080 / script/jquery.js

http:// （协议号）
www （子域名）
google （主域名）
8080 （端口号）
script/jquery.js （请求的地址）

* 当协议、子域名、主域名、端口号中任意一各不相同时，都算不同的“域”。
* 不同的域之间相互请求资源，就叫“跨域”。

出现跨域问题的情况

编号	url	说明	是否允许通信
1	http://www.a.com/a.js http://www.a.com/b.js	同一域名下	允许
2	http://www.a.com/a/a.js http://www.a.com/b/b.js	同一域名不同文件夹	允许
3	http://www.a.com:8080/a.js http://www.a.com:9090/a.js	同一域名不同端口号	不允许
4	http://www.a.com/a.js https://www.a.com/b.js	同一域名不同协议	不允许
5	http://www.a.com/a.js http://192.168.4.158/b.js	域名与域名对应的ip地址	不允许
6	http://www.a.com/a.js http://wwww.a.com/b.js	主域名相同，子域名不同	不允许
7	http://www.a.com/a.js http://www.b.com/b.js	不同域名	不允许

处理跨域的方法

1.JSONP

http://kb.cnblogs.com/page/139725/

在js中，Ajax直接请求普通文件(静态页面、动态网页、web服务、WCF)存在跨域无权限访问的问题。但是，Web页面上调用js文件时则不受是否跨域的影响（凡是拥有“src”这个属性的标签都拥有跨域的能力，比如<script>、<img>、<iframe>），jsonp正是利用这个特性来实现的。

JSONP是一种非官方跨域数据交互协议。解决跨域问题方法的核心是web客户端通过动态添加<script>标签来调用跨域服务器上动态生成的js文件。

该协议的一个要点就是允许用户传递一个callback参数给服务端，然后服务端返回数据时会将这个callback参数作为函数名来包裹住JSON数据，这样客户端就可以随意定制自己的函数来自动处理返回数据了。

比如，有个jsonp.html页面，它里面的代码需要利用ajax获取一个不同域上的json数据，假设这个json数据的地址是http://flightQuery.com/jsonp/flightResult.aspx。

jsonp.html页面的代码：

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <script type="text/javascript">
    // 得到航班信息查询结果后的回调函数
    var flightHandler = function(data){
        alert('你查询的航班结果是：票价 ' + data.price + ' 元，' + '余票 ' + data.tickets + ' 张。');
    };

    // 提供jsonp服务的url地址
    var url = "http://flightQuery.com/jsonp/flightResult.aspx?code=CA1998&callback=flightHandler";
//code参数：告诉服务器，我要查的是CA1998次航班的信息。
//callback参数：告诉服务器，我的本地回调函数就是callback的参数flightHandler，所以请把查询结果传入这个函数中进行调用。

   // 创建script标签，设置其属性
    var script = document.createElement('script');
    script.setAttribute('src', url);

    // 把script标签加入head，此时调用开始
    document.getElementsByTagName('head')[0].appendChild(script);
    </script>
</head>
<body>
</body>
</html>
```

于是flightResult.aspx页面生成了一段JSON格式的javascript代码提供给jsonp.html（服务端的实现忽略）：

```
flightHandler({
    "code": "CA1998",
    "price": 1780,
    "tickets": 5
});
//服务端返回数据时将callback的参数flightHandler作为函数名来包裹JSON数据。这个JSON数据里描述了航班的基本信息。
```

运行一下页面，成功弹出提示窗口，jsonp的执行全过程顺利完成。

用jQuery实现jsonp调用

(1).$.ajax()

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
 <html xmlns="http://www.w3.org/1999/xhtml" >
 <head>
    <title>Untitled Page</title>
     <script type="text/javascript" src=jquery.min.js"></script>
     <script type="text/javascript">
```

```
    jQuery(document).ready(function(){
      $.ajax({
          type: "get",
          async: false,
          url: "http://flightQuery.com/jsonp/flightResult.aspx?code=CA1998",
          dataType: "jsonp",
          jsonp: "callback",//传递给请求处理程序或页面的，用以获得jsonp回调函数名的参数名(一般默认为:callback)
          jsonpCallback:"flightHandler",//自定义的jsonp回调函数名称，默认为jQuery自动生成的随机函数名，也可以写"?"，jQuery会自动为你处理数据
          success: function(json){
              alert('您查询到航班信息：票价： ' + json.price + ' 元，余票： ' + json.tickets + ' 张。');
          },
          error: function(){
              alert('fail');
          }
       });
    });
    </script>
    </head>
  <body>
  </body>
 </html>
```

这里没有写flightHandler这个函数，这是因为jquery在处理jsonp类型的ajax时，自动生成回调函数并把数据取出来供success属性方法来调用。

(2).$.getJSON()

http://www.cnblogs.com/2050/p/3191744.html

```
<script>
$.getJSON('http://example.com/data.php?callback=?',function(jsondata){
    //处理获得的json数据
});
</script>
```

不需要手动的插入script标签以及定义回掉函数。jquery会自动生成一个全局函数来替换callback=?中的问号，之后获取到数据后又会自动销毁，实际上就是起一个临时代理函数的作用。$.getJSON方法会自动判断是否跨域，不跨域的话，就调用普通的ajax方法；跨域的话，则会以异步加载js文件的形式来调用jsonp的回调函数。

ajax和jsonp

(1) ajax和jsonp这两种技术在调用方式上“看起来”很像，目的也一样，都是请求一个url，然后把服务器返回的数据进行处理，因此jquery和ext等框架都把jsonp作为ajax的一种形式进行了封装；

(2) 但ajax和jsonp其实本质上是不同的东西。ajax的核心是通过XmlHttpRequest获取非本页内容，而jsonp的核心则是动态添加<script>标签来调用服务器提供的js脚本。

(3) 所以说，其实ajax与jsonp的区别不在于是否跨域，ajax通过服务端代理一样可以实现跨域，jsonp本身也不排斥同域的数据的获取。

(4) jsonp是一种方式或者说非强制性协议，如同ajax一样，它也不一定非要用json格式来传递数据，如果你愿意，字符串都行，只不过这样不利于用jsonp提供公开服务。

2.服务器端代理

通过后台(ASP、PHP、JAVA、ASP.NET)获取其他域名下的内容，然后再把获得内容返回到前端，这样因为在同一个域名下，所以就不会出现跨域的问题。

比如在A（www.a.com/sever.php）和B（www.b.com/sever.php）各有一个服务器，A的后端（www.a.com/sever.php）直接访问B的服务，然后把获取的响应值返回给前端。也就是A的服务在后台做了一个代理，前端只需要访问A的服务器也就相当与访问了B的服务器。

3 XHR2

“XHR2” 全称 “XMLHttpRequest Level2”，是HTML5提供的方法，对跨域访问提供了很好的支持，并且还有一些新的功能。

只需要在服务器端头部加上下面两句代码：

```
header( "Access-Control-Allow-Origin:*" );
header( "Access-Control-Allow-Methods:POST,GET" );
```

* IE10以下的版本都不支持

4.WebSocket

WebSocket技术是对HTTP无状态连接的一种革新，本质就是一种持久性socket连接，在浏览器客户端通过javascript进行初始化连接后，就可以监听相关的事件和调用socket方法来对服务器的消息进行读写操作。

与Ajax相比，Ajax技术需要客户端发起请求，而WebSocket服务器和客户端可以彼此相互推送信息；XHR受到域的限制，而WebSocket允许跨域通信，这个特性导致我们至少可以用来做远控。

WebSocket实现了全双工通信，使WEB上的真正的实时通信成为可能。浏览器和服务器只需要做一个握手的动作，然后，浏览器和服务器之间就形成了一条快速通道。两者之间就直接可以数据互相传送。

在WebSocket协议中，为我们实现即时服务带来了三个好处：

1. 客户端和服务器端之间数据传输时请求头信息比较小,大概2个字节。
2. 服务器和客户端可以相互主动的发送数据给对方。
3. 不需要多次创建TCP请求和销毁，节约宽带和服务器的资源。

5.cors

http://www.ruanyifeng.com/blog/2016/04/cors.html