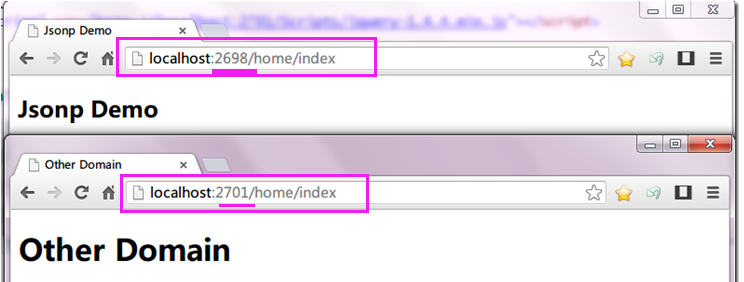
# Jsonp实现跨域

## Jsonp产生的原因

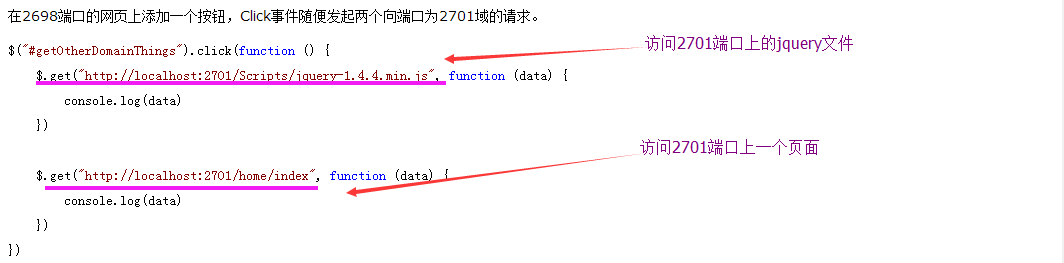
* 浏览器基于安全的原因，存在“同源策略”机制，**同源策略阻止从一个源加载的文档或脚本获取或设置另一个源加载的文档的属性**

例如：

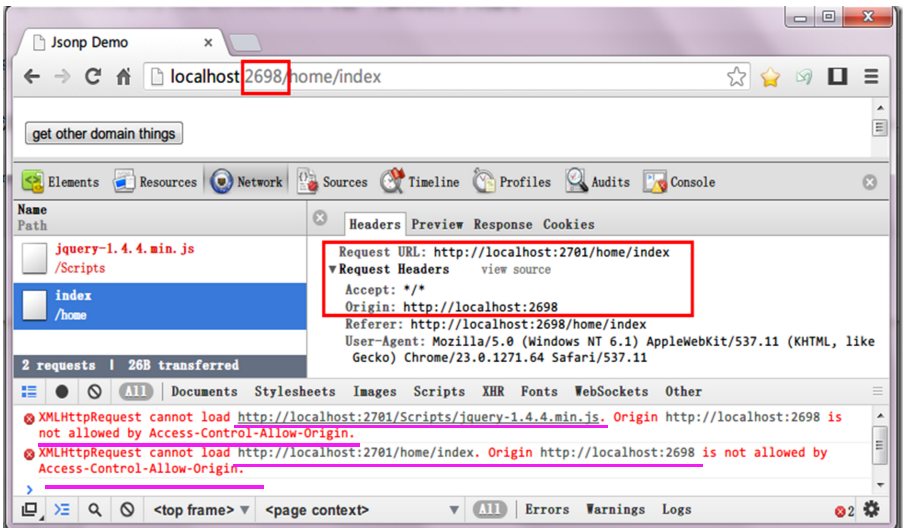
* 两个网页，端口不同，分别是2698、2701，是不同源



* 使用js或者jquery向不同源发出请求



由于同源策略，浏览器是不允许访问的。



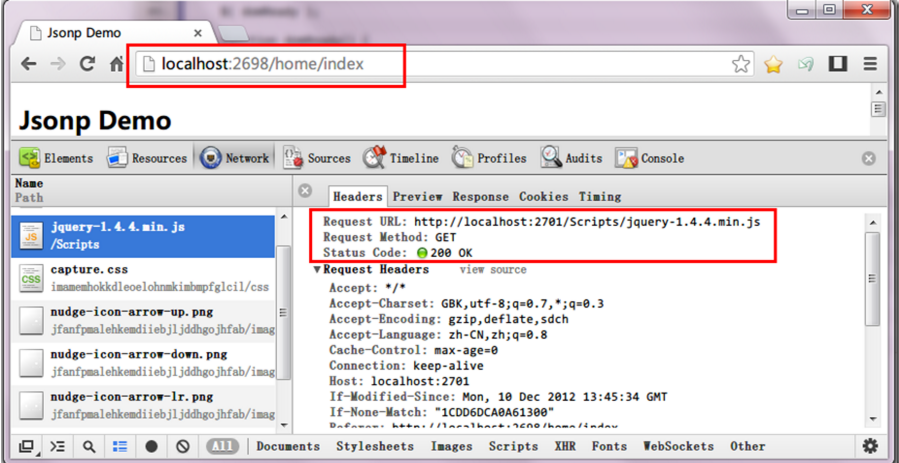
由此，聪明的程序猿就创造了一种非正规的jsonp方法来实现跨域

* **jsonp怎么产生？**

1. 一个众所周知的问题，Ajax直接请求普通文件存在跨域无权限访问的问 题，甭管你是静态页面、动态网页、web服务、WCF，只要是跨域请求， 一律不准；
2. 不过我们又发现，Web页面上调用js文件时则不受是否跨域的影响（不 仅如此，我们还发现凡是拥有"src"这个属性的标签都拥有跨域的能力， 比如<script>、<img>、<iframe>）；
3. 于是可以判断，当前阶段如果想通过纯web端（ActiveX控件、服务端 代理、属于未来的HTML5之Websocket等方式不算）跨域访问数据就 只有一种可能，那就是在远程服务器上设法把数据装进js格式的文件里， 供客户端调用和进一步处理；
4. 恰巧我们已经知道有一种叫做JSON的纯字符数据格式可以简洁的描述 复杂数据，更妙的是JSON还被js原生支持，所以在客户端几乎可以随 心所欲的处理这种格式的数据；
5. 这样子解决方案就呼之欲出了，web客户端通过与调用脚本一模一样的 方式，来调用跨域服务器上动态生成的js格式文件（一般以JSON为后 缀），显而易见，服务器之所以要动态生成JSON文件，目的就在于把 客户端需要的数据装入进去。
6. 客户端在对JSON文件调用成功之后，也就获得了自己所需的数据，剩下 的就是按照自己需求进行处理和展现了，这种获取远程数据的方式看起来 非常像AJAX，但其实并不一样。
7. 为了便于客户端使用数据，逐渐形成了一种非正式传输协议，人们把它称 作JSONP，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。

## 了解script标签的跨域能力

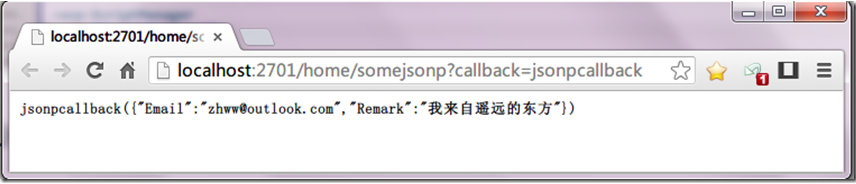
* 在2698端口的一个页面中，可以以script标签的形式，访问2701端口上的jquery文件，浏览器是允许的。



* 因此，可以利用script标签实现跨域，这个也是jsonp的基础，可以在2701端口中创建一个存有2698端口页面需要的数据文件，格式为json，可以在2698端口利用script获取2701这个文件，因为浏览器加载script标签后，是将src存放的文件当成js文件执行，因此文件格式最好是按照合法js语法规范的，否则浏览器会报语法错误。

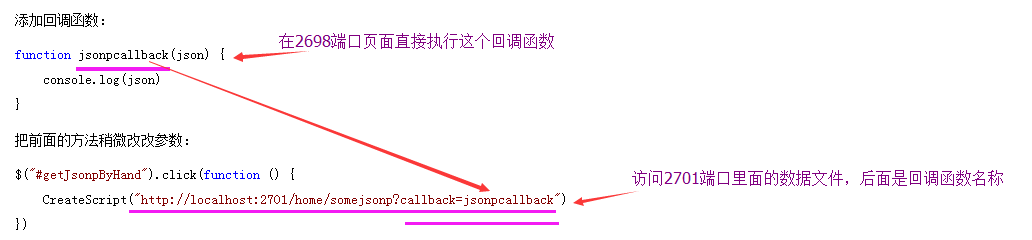
## 利用script获取异域的jsonp

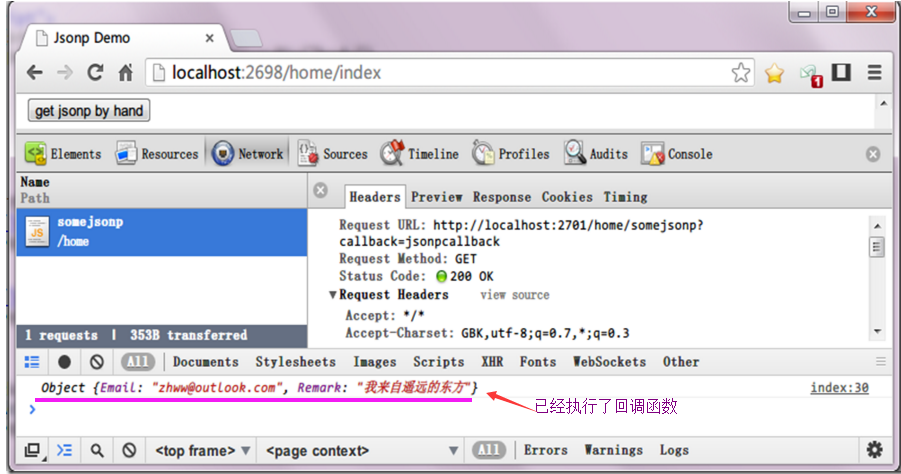
* 把json放到一个回调方法里



如果存在jsonpcallback这个方法，那么

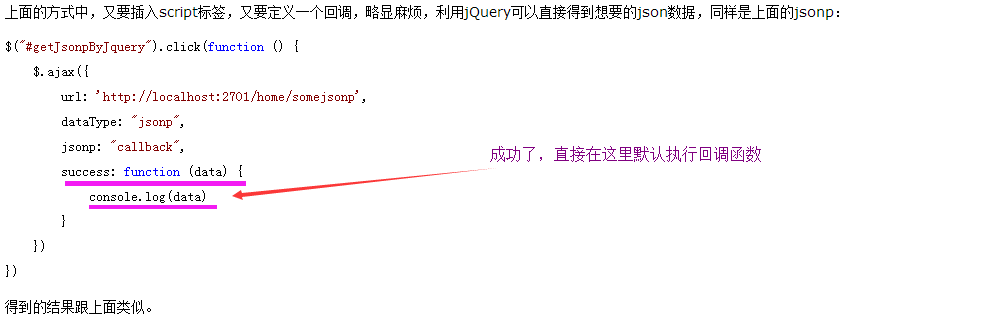
jsonpcallback({"Email":"zhww@outlook.com","Remark":"我来自遥远的东方"})就是合法的js语句。由于服务器不知道客户端的回调是什么，不可能hard code成jsonpcallback，所以就带一个QueryString让客户端告诉服务端，回调方法是什么，当然，QueryString的key要遵从服务端的约定，上面的是”callback“。





## 利用jquery获取jsonp

* 不用插入script标签，也不用定义回调函数

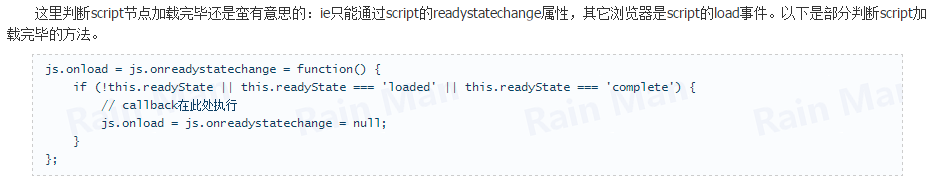


## 总结

* 总之就是利用script标签绕过同源策略，获得一个类似这样的数据，jsonpcallback是页面存在回调方法，参数就是想得到的json

## 六、Jsonp的注意事项

* 使用jsonp跨域需要动态创建标签script，创建使用后，是要将这些节点删除的，只删除script标签是不够的，需要手动清除这段js占用的内存。
* 在js实现jsonp时，动态生成script标签，可以通过一些事件来判断script节点是否加载完毕：



## JSONP的客户端具体实现

* **远程服务器remoteserver.com根目录下有一个remote.js文件，文件代码如下：**



* **本地服务器localserver.com下有一个jsonp.html页面，代码如下：**



在运行此页面时，页面将会弹出一个提示框，显示跨域成功

* **现在在jsonp.html页面定义一个函数，然后在远程remote.js中传入数据进行调用**

Jsonp.html页面代码如下：



Remote.js文件代码：



**注：运行之后查看结果，页面成功弹出提示窗口，显示本地函数被跨域的远程js调用 成功，并且还接收到了远程js带来的数据。很欣喜，跨域远程获取数据的目的基本实 现了，但是又一个问题出现了，我怎么让远程js知道它应该调用的本地函数叫什么名 字呢？毕竟是jsonp的服务者都要面对很多服务对象，而这些服务对象各自的本地函 数都不相同啊？我们接着往下看。**

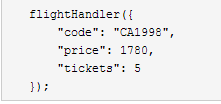
* 只要服务端提供的js脚本是动态生成的就行了呗，这样调用者可以传一个参数过去告诉服务端“我想要一段调用XXX函数的js代码，请你返回给我”，于是服务器就可以按照客户端的需求来生成js脚本并响应了。

Jsonp.html页面代码：



**注：**这次的代码变化比较大，不再直接把远程js文件写死，而是编码实现动态查询， 而这也正是jsonp客户端实现的核心部分。我们看到调用的url中传递了一个code参 数，告诉服务器我要查的是CA1998次航班的信息，而callback参数则告诉服务器， 我的本地回调函数叫做flightHandler，所以请把查询结果传入这个函数中进行调用。

* 这个叫做flightResult.aspx的页面生成了一段这样的代码提供给jsonp.html（服务端的实现这里就不演示了，与你选用的语言无关，说到底就是拼接字符串）：



**结果：**传递给flightHandler函数的是一个json，它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！最后就是如何把代码封 装 一下，以便于与用户界面交互，从而实现多次和重复调用。

* Jquery实现jsonp调用，延续上面航班信息查询的例子：假定返回的jsonp结果不变：



* Ajax与jsonp的区别：

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. 还有就是，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。
5. 总之：jsonp不是ajax的一个特例