彻底弄懂jsonp原理及实现方法

**一、 同源策略**

所有支持Javascript的浏览器都会使用同源策略这个安全策略。看看百度的解释：

同源策略，它是由Netscape提出的一个著名的安全策略。

现在所有支持JavaScript 的浏览器都会使用这个策略。所谓同源是指，域名，协议，端口相同。

当一个浏览器的两个tab页中分别打开百度和谷歌的页面

当一个百度浏览器执行一个脚本的时候会检查这个脚本是属于哪个页面的

即检查是否同源，只有和百度同源的脚本才会被执行。

而解决这种同源策略的方法称之为跨域

跨域的方法有很多种,这里介绍一下最经典的jsonp跨域。

### 二、JSON和JSONP

JSONP和JSON好像啊，他们之间有什么联系吗？

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。对于JSON大家应该是很了解了吧，不是很清楚的朋友可以去[json.org](http://www.json.org/json-zh.html" \t "_blank)上了解下，简单易懂。

JSONP是JSON with Padding的略称。它是一个非官方的协议，它允许在服务器端集成Script tags返回至客户端，通过javascript callback的形式实现跨域访问（这仅仅是JSONP简单的实现形式）。--来源百度

　　JSONP就像是JSON+Padding一样(Padding这里我们理解为填充)， 我们先看下面的小例子然后再详细介绍。

### 三、跨域的简单原理

光看定义还不是很明白，那首先我们先来手动做个简单易懂的小测试。新建一个asp.net的web程序，添加**sample.html**网页和一个**test.js**文件，代码如下：

sample.html的代码：

//sample.html

<!DOCTYPE>

<html>

<head>

<title>test</title>

<script type="text/javascript" src="test.js"></script>

</head>

<body>

</body>

</html>

test.js的代码：  
//test.js

alert("success");

打开sample.html后会跳出"success”这样的这样的信息框，这似乎并不能说明什么， 跨域问题到底怎么解决呢？

好，现在我们模拟下非同源的环境，刚才我们不是已经用Visual Studio新建了一个Web程序吗(这里我们叫A程序)，现在我们再打开一个新的Visual Studio再新建一个Web程序(B程序)，将我们的之前的test.js文件从A程序中移除然后拷贝到B程序中。

两个程序都运行起来后，Visual Studio会启动内置服务器，假设A程序是localhost:20001,B程序是localhost:20002，这就模拟了一个非同源的环境了(虽然域名相同但端口号不同，所以是非同源的)。

OK，我们接下来应该改下sample.html里的代码，因为test.js文件在B程序上了，url也就变成了localhost:20002。

sample.html部分代码:

<script type="text/javascript" src="http://localhost:20002/test.js"></script>

请保持AB两个Web程序的运行状态，当你再次刷新localhost:20001/sample.html的时候，和原来一样跳出了"success"的对话框.

是的，成功访问到了非同源的localhost:20002/test.js这个所谓的远程服务了。到了这一步，相信大家应该已经大概明白如何跨域访问了的原理了。

<script>标签的src属性并不被同源策略所约束，所以可以获取任何服务器上脚本并执行。

### 四、JSONP的实现模式--CallBack

刚才的小例子讲解了跨域的原理，我们回上去再看看JSONP的定义说明中讲到了javascript callback的形式。那我们就来修改下代码，如何实现JSONP的javascript callback的形式。

程序A中sample的部分代码：

<script type="text/javascript">

//回调函数

function callback(data) {

alert(data.message);

}

</script>

<script type="text/javascript" src="http://localhost:20002/test.js"></script>

程序B中test.js的代码：

//调用callback函数，并以json数据形式作为阐述传递，完成回调

callback({message:"success"});

这其实就是JSONP的简单实现模式，或者说是JSONP的原型：创建一个回调函数，然后在远程服务上调用这个函数并且将JSON 数据形式作为参数传递，完成回调。

将JSON数据填充进回调函数，这就是JSONP的JSON+Padding的含义吧。

一般情况下，我们希望这个script标签能够动态的调用，而不是像上面因为固定在html里面所以没等页面显示就执行了，很不灵活。我们可以通过javascript动态的创建script标签，这样我们就可以灵活调用远程服务了。

程序A中sample的部分代码：

<script type="text/javascript">

function callback(data) {

alert(data.message);

}

//添加<script>标签的方法

function addScriptTag(src){

var script = document.createElement('script');

script.setAttribute("type","text/javascript");

script.src = src;

document.body.appendChild(script);

}

window.onload = function(){

addScriptTag("http://localhost:20002/test.js");

}

</script>

程序B的test.js代码不变，我们再执行下程序，是不是和原来的一样呢。

如果我们再想调用一个远程服务的话，只要添加addScriptTag方法，传入远程服务的src值就可以了。

这里说明下为什么要将addScriptTag方法放入到window.onload的方法里，原因是addScriptTag方法中有句document.body.appendChild(script);，这个script标签是被添加到body里的，由于我们写的javascript代码是在head标签中，document.body还没有初始化完毕呢，所以我们要通过window.onload方法先初始化页面，这样才不会出错。

上面的例子是最简单的JSONP的实现模型，不过它还算不上一个真正的JSONP服务。

我们来看一下真正的JSONP服务是怎么样的，比如Google的ajax搜索接口：[http://ajax.googleapis.com/ajax/services/search/web?v=1.0&q=?&callback=](http://ajax.googleapis.com/ajax/services/search/web?v=1.0&q=?&callback=" \t "_blank)?

q=?这个问号是表示你要搜索的内容，最重要的是第二个callback=?这个是正如其名表示回调函数的名称，也就是将你自己在客户端定义的回调函数的函数名传送给服务端，服务端则会返回以你定义的回调函数名的方法，将获取的json数据传入这个方法完成回调。有点罗嗦了，还是看看实现代码吧:

<script type="text/javascript">

//添加<script>标签的方法

function addScriptTag(src){

var script = document.createElement('script');

script.setAttribute("type","text/javascript");

script.src = src;

document.body.appendChild(script);

}

window.onload = function(){

//搜索apple，将自定义的回调函数名result传入callback参数中

addScriptTag("http://ajax.googleapis.com/ajax/services/search/web?v=1.0&q=apple&callback=result");

}

//自定义的回调函数result

function result(data) {

//我们就简单的获取apple搜索结果的第一条记录中url数据

alert(data.responseData.results[0].unescapedUrl);

}

</script>

### 五、jQuery对JSONP的实现

jQuery框架也当然支持JSONP，可以使用$.getJSON(url,[data],[callback])方法(详细可以参考JQ实现jsonp)。

那我们就来修改下程序A的代码，改用jQuery的getJSON方法来实现(下面的例子没用用到向服务传参，所以只写了getJSON(url,[callback]))：

<script type="text/javascript" src="http://code.jquery.com/jquery-latest.js"></script>

<script type="text/javascript">

$.getJSON("http://localhost:20002/MyService.ashx?callback=?",function(data){

alert(data.name + " is a a" + data.sex);

});

</script>

结果是一样的，**要注意的是在url的后面必须添加一个callback参数，这样getJSON方法才会知道是用JSONP方式去访问服务**，callback后面的那个问号是内部自动生成的一个回调函数名。这个函数名大家可以debug一下看看，比如jQuery17207481773362960666\_1332575486681。

当然，加入说我们想**指定自己的回调函数名**，或者说服务上规定了固定回调函数名该怎么办呢？我们可以使用$.ajax方法来实现(参数较多，详细可以参考http://api.jquery.com/jQuery.ajax)。先来看看如何实现吧：

没错，jsonpCallback就是可以指定我们自己的回调方法名person，远程服务接受callback参数的值就不再是自动生成的回调名，而是person。dataType是指定按照JSOPN方式访问远程服务。

利用jQuery可以很方便的实现JSONP来进行跨域访问。先暂时写到这吧。

博文引用自：

https://www.cnblogs.com/soyxiaobi/p/9616011.html