# 八 跨域

### 1 跨域的产生

在访问文件时，只有同域名下的不同文件夹下的资源可以互相访问，不同的IP、端口、协议、二级域名、带3w和不带3w之间都是不允许通信的，这时候就会产生跨域请求，即：

如果一个域名的下文件请求了和它不一样的域名下的资源，就会产生跨域请求。

比如：一个接口地址，我们在浏览器直接输入可以获取数据，而，在本地localhost环境下使用ajax请求却无法获得数据。

跨域解决方案：

1 iframe：包含跨域的文件，但无法对其内部进行dom操作、处理数据

2 后台请求：比如让本地php请求跨域资源，然后ajax访问本地的php；

3 Flash

4 JSONP：json with padding

注意：

顶级域名相同的可以通过domain.name来解决，即同时设置 domain.name = 顶级域名（如example.com）

document.domain + iframe

window.name + iframe

location.hash + iframe

window.postMessage()

### 2 JSONP

JSON with Padding其本质是利用了<script src=""></script>标签具有可跨域的特性，由服务端返回一个预先定义好的Javascript函数的调用，并且将服务器数据以该函数参数的形式传递过来，此方法需要前后端配合完成。

比如我们常见的直接使用script标签引入cdn上的jquery文件就是利用了script标签的跨域功能。

那么这时，我们直接使用script标签来引入接口地址，也能获取到接口返回的数据，但是遗憾的是，这个数据没有变量名保存，我们如何使用该数据呢？

很显然，我们需要解决script标签加载过来的json数据无法通过变量保存后来使用的问题。

解决方案一：返回的数据用变量名赋值。

var a = [1,2,3];

通过script标签加载后，就会在js文件中声明了一个变量a，这时候可以直接使用数据了，但是缺点也很明显，多了一个全局变量，所以这种做法不好。

解决方案二：返回的数据用函数包裹起来：

fn([1,2,3]);

<script>  
 function fn(data) {  
 console.log(data);  
 }  
</script>  
<script src="1.txt"></script>

步骤：

    在资源加载进来之前定义一个函数，这个函数接收一个参数（数据），函数里利用这个参数进行数据处理；

    通过script标签加载对应的远程文件资源，当远程文件资源被加载进来的时候，就会去执行前面定义好的函数，并且把资源数据当做参数传入。

此时仍然有问题，因为src加载文件是打开网页就会立即执行了该函数，我们需要按需加载。那么这里我们可以使用动态创建script标签的形式来解决该问题。

前端代码:注意,域名不同，核心是 通过script标签的src属性提交get请求

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Document</title>

<script type="text/javascript">

function fn(data){

console.log(data);

}

</script>

<script type="text/javascript" src='http://www.section02.com/seciton02\_jsonP.php?callback=fn'>

</script>

</head>

<body>

<h1>区域1的页面\_jsonP演示</h1>

</body>

</html>

后台代码：

<?php

// echo "alert('天气不错哦')";

$callBack = $\_GET['callback'];

$arr = array(

'name' =>'西兰花' ,

'color' =>'red'

);

echo $callBack."(".json\_encode($arr).")";

?>

### 3 jQuery中的跨域

jQuery方法当中集成了JSONP的实现，可以非常方便的实现跨域数据的访问。

dataType: 'jsonp' 设置dataType值为jsonp即开启跨域访问

jsonp 可以指定服务端接收的参数的“key”值，默认为callback

jsonpCallback 可以指定相应的回调函数，默认自动生成

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Document</title>

<script type="text/javascript" src='jquery/jquery-3.0.0.min.js'></script>

</head>

<body>

<h1>区域1的页面</h1>

<input type="button" name="" onclick='sendAjax()' value="jquery区域请求">

</body>

</html>

<script type="text/javascript">

function sendAjax(){

$.ajax({

url:'http://www.section02.com/sectcion02\_jqJsonp.php',

type:'post',

dataType: 'jsonp',

data:{name:'itt'},

success:function(data){

console.log(data);

}

})

}

</script>

jsonp案例---百度搜索

<!DOCTYPE HTML>  
<html>  
<head>  
 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  
 <title>无标题文档</title>  
 <style>  
 #q {width: 300px; height: 30px; padding: 5px; border:1px solid #f90; font-size: 16px;}  
 #ul1 {border:1px solid #f90; width: 310px; margin: 0;padding: 0; display: none;}  
 li a { line-height: 30px; padding: 5px; text-decoration: none; color: black; display: block;}  
 li a:hover{ background: #f90; color: white; }  
 </style>  
 <script>  
 function maiov(data) {  
  
 var oUl = document.getElementById('ul1');  
 var html = '';  
 if (data.s.length) {  
 oUl.style.display = 'block';  
 for (var i=0; i<data.s.length; i++) {  
 html += '<li><a target="\_blank" href="http://www.baidu.com/s?wd='+data.s[i]+'">'+ data.s[i] +'</a></li>';  
 }  
 oUl.innerHTML = html;  
 } else {  
 oUl.style.display = 'none';  
 }  
  
 }  
 window.onload = function() {  
  
 var oQ = document.getElementById('q');  
 var oUl = document.getElementById('ul1');  
  
 oQ.onkeyup = function() {  
  
 if ( this.value != '' ) {  
 var oScript = document.createElement('script');  
 oScript.src = 'http://suggestion.baidu.com/su?wd='+this.value+'&cb=maiov';  
 document.body.appendChild(oScript);  
 } else {  
 oUl.style.display = 'none';  
 }  
  
 }  
  
 }  
 </script>  
</head>  
  
<body>  
<input type="text" id="q" />  
<ul id="ul1"></ul>  
</body>  
</html>

豆瓣API：

<!DOCTYPE HTML>  
<html>  
<head>  
 <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  
 <title>无标题文档</title>  
 <style>  
 #q {width: 300px; height: 30px; padding: 5px; border:1px solid #f90; font-size: 16px;}  
 dl {border-bottom: 1px dotted #000;}  
 dt {font-weight: bold;}  
 </style>  
 <script>  
 function fn1(data) {  
  
 var oMsg = document.getElementById('msg');  
 var oList = document.getElementById('list');  
  
 console.log(data);  
  
 oMsg.innerHTML = data.title.$t + ' : ' + data['opensearch:totalResults'].$t;  
  
 var aEntry = data.entry;  
 var html = '';  
 for (var i=0; i<aEntry.length; i++) {  
  
 html += '<dl><dt>'+ aEntry[i].title.$t +'</dt><dd><img src="'+ aEntry[i].link[2]['@href'] +'" /></dd></dl>';  
  
 }  
  
 oList.innerHTML = html;  
  
 }  
 window.onload = function() {  
  
 var oQ = document.getElementById('q');  
 var oBtn = document.getElementById('btn');  
 var oMsg = document.getElementById('msg');  
 var oList = document.getElementById('list');  
  
 oBtn.onclick = function() {  
  
 if ( oQ.value != '' ) {  
 var oScript = document.createElement('script');  
 oScript.src = 'http://api.douban.com/book/subjects?q='+oQ.value+'&alt=xd&callback=fn1';  
 document.body.appendChild(oScript);  
 }  
  
 *//http://api.douban.com/book/subjects?q='+oQ.value+'&alt=xd&callback=fn1&start-index=(当前页\*每页显示的条数)&max-results=10(每页显示的条数)* }  
  
 }  
 </script>  
</head>  
  
<body>  
http://www.douban.com/service/apidoc/reference/  
<input type="text" id="q" /><input type="button" id="btn" value="搜索" />  
<p id="msg"></p>  
<hr />  
<div id="list"></div>  
</body>  
</html>

# 九 Ajax2.0

## 1 FormData

Ajax2.0在文件上传、跨域上有很优秀的体验。（websocket也支持文件上传、跨域，但是非常麻烦）。

Ajax2.0使用FormData 对象控制提交数据、文件上传等，常见的方法：

set(key,value);

get(key,value);

delete(key,value);

getAll(key);

比如一个表单

<form> <input type=”text” name=”user” value=”lisi”> </form>

在Ajax2.0中，我们这样书写：

let data = new FormData();

data.set(‘user’,’lisi’);

注意：

多次书写set会覆盖之前的值。那么如何提交1个user，2个值的情况？

在以前可以在form表单中连续书写两个input，名字都为user，值不同，提交给服务器。formdata提供了 append() 方法来解决：

data.append(‘user’,zs);

注意：FormData最常用来上传文件

## 2 示例

用户名：<input type="text" name="username" id="username"><br>  
密 码：<input type="text" name="password" id="password"><br>  
<input type="submit" value="提交" id="btn">  
<script>  
  
 let username = document.querySelector('#username');  
 let password = document.querySelector('#password');  
 document.querySelector('#btn').onclick = function () {  
  
 let data = new FormData();  
 data.set('username',username.value);  
 data.set('password',password.value);  
  
 let ajax = new XMLHttpRequest();  
  
/\*  
 //GET  
 let arr = [];  
 data.forEach((value,key)=>{  
 arr.push(`${encodeURIComponent(key)}=${encodeURIComponent(value)}`);  
 });  
 let path = `http://localhost:8000/api?${arr.join('&')}`;  
 ajax.open('GET',path,true);  
 ajax.send();  
\*/  
  
 *//POST* let path = `http://localhost:8000/api`;  
 ajax.open('POST',path,true);  
 ajax.setRequestHeader('Content-Type','application/x-www-form-urlencoded');  
 ajax.send(data);  
  
 ajax.onreadystatechange = function () {  
 if(ajax.readyState ==4) {  
 if(ajax.status >= 200 && ajax.status < 300 || ajax.status == 304){  
 console.log('sucess');  
 } else {  
 console.log('err');  
 }  
 }  
 }  
  
 }

如果我们在服务器端打印POST请求提交的data，会发现并没有按 application/x-ww-form-urlencoded这种普通提交方式提交，而是仍然以上传文件的形式得到了data。所以这句设置没有多少意义！

## 3 FormData的使用场景-上传

FormData上传无需刷新页面，具体代码见 Node文档

## 4 cros跨域