**关于jsonp的认识与理解**

1. **JSONP的概念**

全称：Javascript Object Notation with Padding(json数据包装)。通俗的理解：jsonp就是一段js代码，使用script标签来引用的，将json数据包装到一个js函数。而json是一种数据格式。

1. **JSONP的作用**

可以让网页从别的域名（网站）获取资料，用于传输从其他域下返回的jsonp数据，简单的理解：跨域读取数据。

跨域：从一个平台到另一平台。如：windows到linux，或windows到另一外一台windows.

1. **JSONP的原理**

网络受同源策略影响，不允许进行跨域请求的，如ajax请求。而script检签src属性中的链接却可以访问跨域的js脚本，利用这个特性，服务端不再返回json格式的数据，而是返回一段调用某一个函数的js代码（callbackfoo({key:value,key1:value1})），在src中进行调用，这样实现了跨域。

1. **JSONP的使用**
2. 可以在js下使用。

新建一个jsonp.php文件，

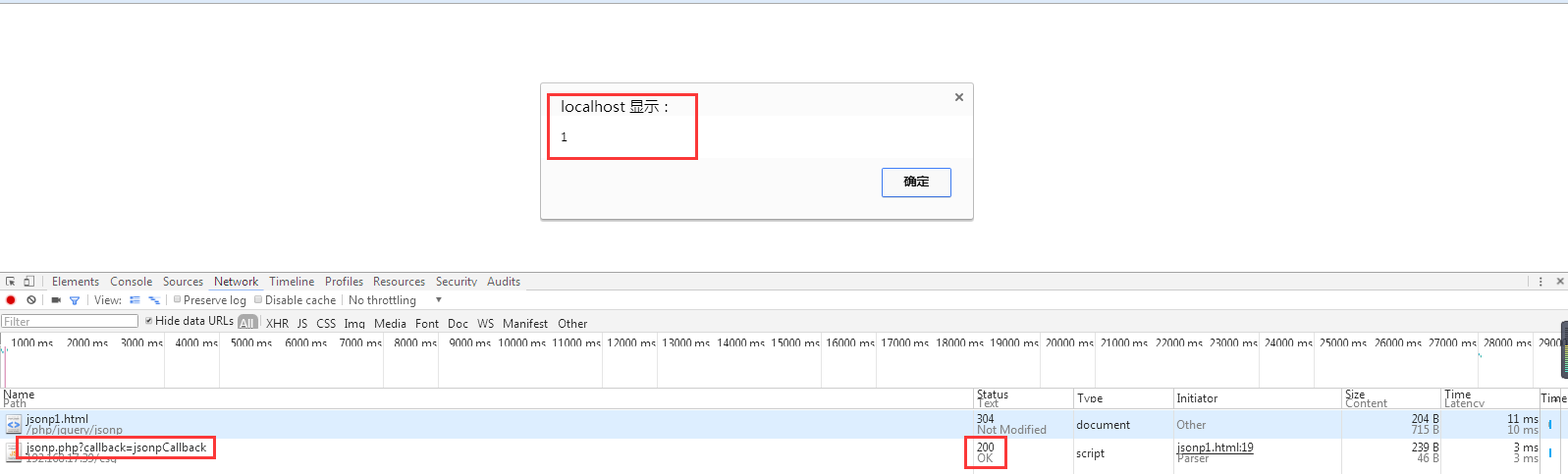
<?php *//服务端返回JSON数据*$arr=array('a'=>1,'b'=>2,'c'=>3,'d'=>4,'e'=>5);  
$result=json\_encode($arr);  
*//echo $\_GET['callback'].'("Hello,World!")';  
//echo $\_GET['callback']."($result)";  
//动态执行回调函数*$callback=$\_GET['callback'];  
echo $callback."($result)";

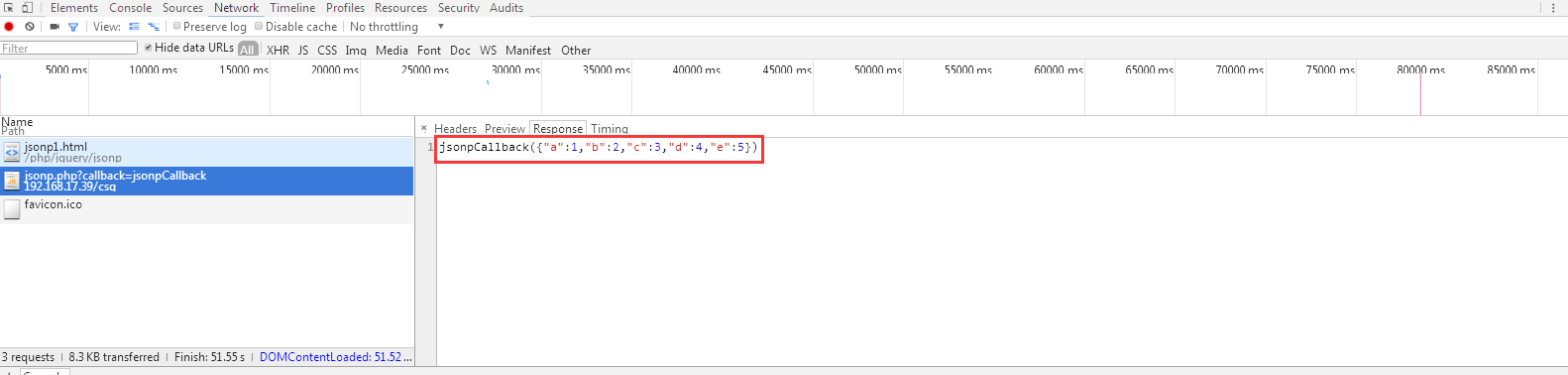
?>

新建一个jsonp.php

<!DOCTYPE html>  
<html lang=**"en"**>  
<head>  
 <meta charset=**"UTF-8"**>  
 <title>**jsonp1的简单使用**</title>  
 *<!--<script src="../js/jquery-3.2.1.min.js"></script>-->* <script type=**"text/javascript"**>  
 function *jsonpCallback*(result) {  
 alert(result.a);  
*// console.log(result);  
// alert(result.message);  
// alert(result.b);  
// alert(result.c);  
// for(var i in result) {  
// alert(i+":"+result[i]);//循环输出a:1,b:2,etc.  
// }* }  
 </script>  
 <script type=**"text/javascript"** src=**"http://192.168.17.39/csg/jsonp.php?callback=jsonpCallback"**></script>  
</head>  
<body>  
</body>  
</html>

运行结果截图：





返回结果：jsonpCallback({"a":1,"b":2,"c":3,"d":4,"e":5})

1. 在jQuery下使用。

新建一个jsonp.php文件

<?php

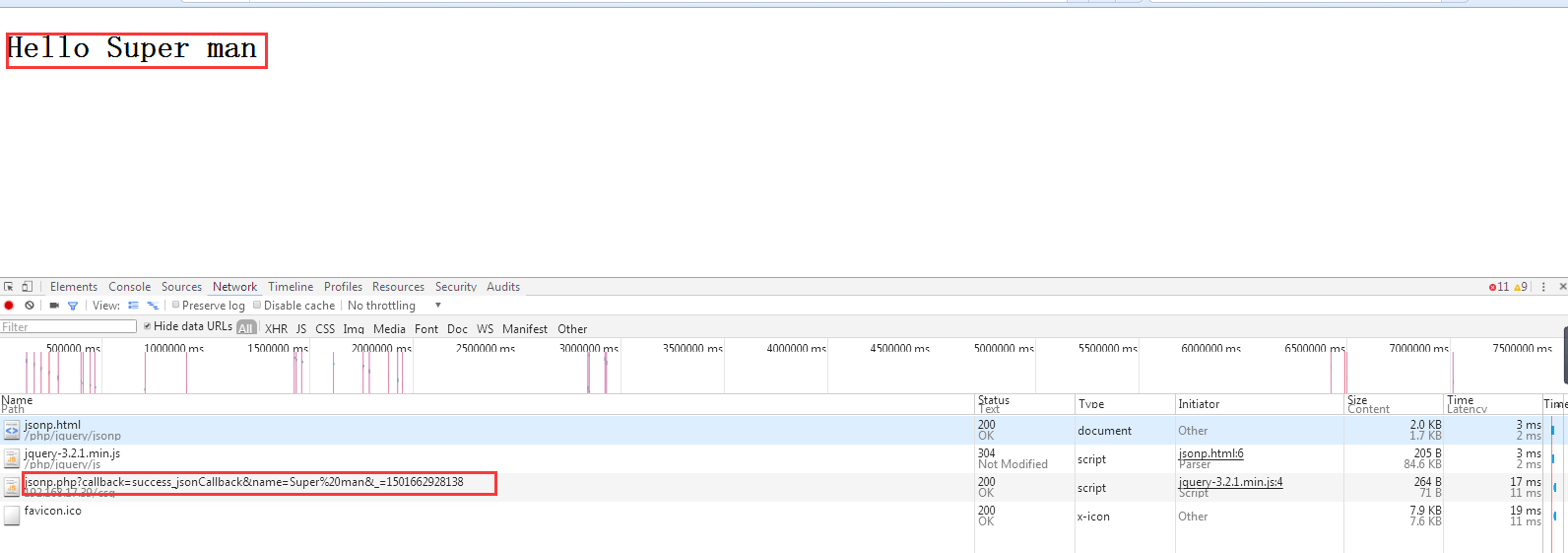
if(isset($\_GET['name']) && isset($\_GET['callback'])) {  
 $obj->name = $\_GET['name'];  
 $obj->message = "Hello " . $obj->name;  
  
 echo $\_GET['callback']. '(' . json\_encode($obj) . ');'; }

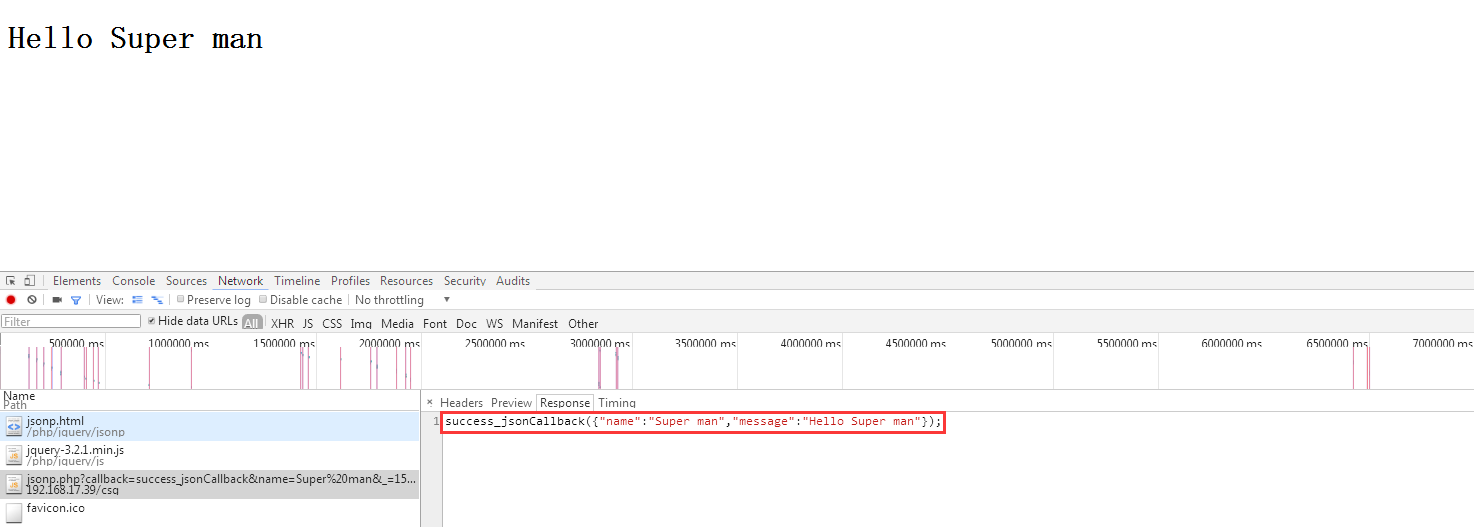
?>

新建一个jsonp.html文件：

<!DOCTYPE html>  
<html lang=**"en"**>  
<head>  
 <meta charset=**"UTF-8"**>  
 <title>**jsonp的简单使用**</title>  
 <script src=**"../js/jquery-3.2.1.min.js"**></script>  
 <script type=**"text/javascript"**>  
 $(document).ready(function () {  
 $.ajax({  
*// url: 'http://localhost/php/jquery/jsonp/jspon.php',* data:{name:'Super man'},  
 dataType:'jsonp', *// tried json* url:'http://192.168.17.39/csg/jsonp.php',  
 type:'GET',  
 jsonp:'callback' ,*// tried true* jsonpCallback:'success\_jsonCallback'  
 *//共有两种响应的方式，要么用success,要么用新建的回调函数。  
// success: function(response) {  
// $("body").html('<h1>' + response.a + '</h1>');  
//// console.log(response.message);  
//// alert(response.message);  
// }* });  
  
 });  
  
 *//3种方法的不同写法  
 // 1* window.success\_jsonCallback=function(data) {  
*// console.log(data.name);* $("body").append('<h1>' + data.message + '</h1>');  
 };  
  
  
 *// 2  
// var success\_jsonCallback = function(saveData) {  
// console.log(saveData);  
// if (saveData == null) {  
// alert("DATA IS UNDEFINED!"); // displays every time  
// }  
// alert("Success is " + saveData.message); // 'Success is undefined'  
// };  
  
// // 3  
// function success\_jsonCallback(data) {  
// alert(data.message);  
// }* </script>  
</head>  
<body>  
  
</body>  
</html>

结果截图：





返回结果：success\_jsonCallback({"name":"Super man","message":"Hello Super man"});

**附录：**

**Ajax XMLHttpRequest(XHR)的状态码：**

0：未初始化，还没有调用send()

1：载入，已调用send()方法

2：载入完成，send()方法执行完成

3：交互：正在解析响应内容

4：完成：响应内容完成，可以在客户端调用。

在jQuery下$.ajax的调试：

error:function(XMLHttpRequest,textStatus,errorThrow) {

alert(XMLHttpRequest.status);

alert(XMLHttpRequest.readyStatus);

alert(textStatus);

}

**Ajax Status的状态码：**

1XX：信息提示

2XX：成功（200）

3XX：重定向（304）

4XX：客户端错误（404）

5XX：服务器错误（505）

**jQuery：append() 与 html()**

append()：在原有的基础上插入数据

html()：替换当前的所有的内容。

**jQuery：serialize()与serilizeArray()**

serialize()：将对象转换为字符串

serializeArray()：将对象转换成数组。

**jQuery遍历JSON数组的2中方法**

1.$.each(obj,function(n,value)){}

2.for(var i in obj){}

**Js,jQuery格式化json数据**

js：JSON.parse(),eval(),eval(“(”+str+”)”);

jQuery：$.parseJOSN(jsonText)