<http://www.cnblogs.com/teach/p/5888958.html>

**解决项目中的跨域操作问题**

浏览器存在许多安全策略，其中同源策略就是其中一个，所谓同源策略也叫同域名策略，即只有协议+域名+端口一致的情况下才可以相互访问，其目的就是为了保护用户信息的安全，同源策略现在的范围包括三方面：1）、Cookie、LocalStorage、IndexDB无法读取；2）、DOM无法获取；3）、AJAX请求不能发送。这里主要介绍两种解决AJAX请求不能发送的解决方案：JSONP和CORS。JSONP是一种前端的解决方式；CORS是跨域资源共享，在服务端实现。

**一、JSONP**

 这里使用jquery的$.ajax()方法来实现JSOP跨域访问，其代码如下，

$(document).ready(function() {
$('#btn').on('click', function() {
$.ajax('http://localhost:8081/springmvc/my/my2/1/1',
{
dataType : 'jsonp',
type : 'get',
jsonpCallback: "aa",
success : function(data,status) {
console.log('111:'+data.Hello+' '+data);
}
})
})
})

这里使用jquery通过给一个按钮注册一个点击事件来触发跨域操作，具体看ajax()函数中的参数，第一个指定了一个URL（restful的形式），然后是必须指定dataType为jsonp，指定请求的类型为get，接着jsonCallBack参数指定为aa，这里这个参数的键及键值都可以随便指定，然后是成功之后的回调函数，实现了控制台打印。

下面是后台代码：

@RequestMapping("my2/{courseId}/{name}")

@ResponseBody

public String method2(@PathVariable("courseId") String courseId,@PathVariable("name") String name){

//必须是这种形式的字符串，其中小括号内的字符串必须是标准的json格式的

String str="aa({\"Hello\":\"World\"})";

return str;

}

这里简单的返回了一个字符串，可以看到字符串的形式比较特别，会发现很像JavaScript中的函数调用（aa({"Hello":"World"})），函数aa的形参是一个对象，恭喜你，这里确实是函数的调用，且函数名称必须和前边的jsonCallBack参数定义的值一致，在前端请求的时候的参数jsonCallBack的值即是这里的函数名。

其实jsonp的方式就是会动态的执行js代码，从而实现跨域。

这种方式必须是get方式的请求，且在后台需要动态获得一个参数的值，即jsonCallBack的值，获得此参数的值之后和要返回的字符串进行拼接；如果要传的参数比较多，这种方式就可以了。

**二、CORS**

CORS是跨域资源共享，通过设置响应头的Access-Control-Allow-Origin属性，从而控制跨域资源访问，通常Access-Control-Allow-Origin的值可以设为“\*”，代表对所有的域都可以实现跨域，也可以单独设置为某个域，如http://10.10.18.98:9090，则表示这个域可以跨域访问。

由于需要给响应设置Access-Control-Allow-Origin属性，这里使用filter，代码如下

package com.cn.my.filter;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class CORSFilter implements Filter {

@Override

public void destroy() {

// TODO Auto-generated method stub

}

//实现向响应头部添加Access-Control-Allow-Origin属性，实现跨域访问。

@Override

public void doFilter(ServletRequest sRequest, ServletResponse sResponse,

FilterChain chain) throws IOException, ServletException {

// TODO Auto-generated method stub

HttpServletRequest request=(HttpServletRequest)sRequest;

HttpServletResponse response=(HttpServletResponse)sResponse;

response.setHeader("Access-Control-Allow-Origin", "\*");

chain.doFilter(sRequest, sResponse);

}

@Override

public void init(FilterConfig arg0) throws ServletException {

// TODO Auto-generated method stub

}

}

以上是一个filter，可以对所有的域进行跨域访问本域。下面是filter的配置，

<!--跨域访问-->
<filter>
<filter-name>CORS</filter-name>
<filter-class>com.cn.my.filter.CORSFilter</filter-class>
</filter>
<filter-mapping>
<filter-name>CORS</filter-name>
<url-pattern>/\*</url-pattern>
</filter-mapping>

以上是在web.xml中进行的filter的配置，配置了服务端的跨域访问，在前端可以使用普通的ajax进行访问，

$('#btn').on('click', function() {
$.ajax('http://localhost:8081/springmvc/my/my2/1/1',
{
type : 'get',
success : function(data,stats) {
console.log('111:'+data.sex+' '+data);
}
})
})

复制代码

上面的ajax请求是一般的请求，，同样可以获得后台的值，且后台代码不需要返回特定的字符串，后台只需要返回JSON即可。如下，

@RequestMapping("my2/{courseId}/{name}")

@ResponseBody

public Map<String, String> method2(@PathVariable("courseId") String courseId, @PathVariable("name") String name) {

HashMap<String, String> map = new HashMap<String, String>();

map.put("name", "刘三");

map.put("sex", "女");

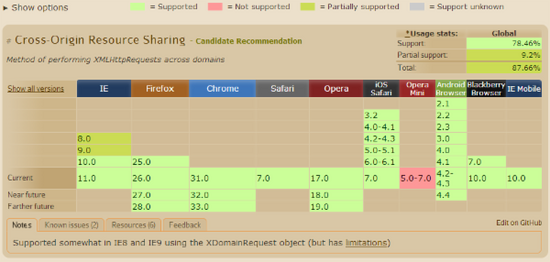
map.put("age", "47");

return map;

}

上面是返回json的后台方法，没有任何的改变。

使用CORS的方式在IE10之下的浏览器是不支持的，下面是一张从网上download的图，可以看到CORS的浏览器的支持情况，



通过两种不同方式的比较可以看出，JSONP对老浏览器支持比较好，但使用起来稍微麻烦，CORS的方式配置简单，只需要一个Filter即可，但是需要较新的浏览器，值得庆幸的是，现在大多数浏览器都支持了CORS。