**SpringBoot 中到底如何解决跨域问题？**

<https://mp.weixin.qq.com/s/bQcPyoHrgaNJqhbb36gQHw>

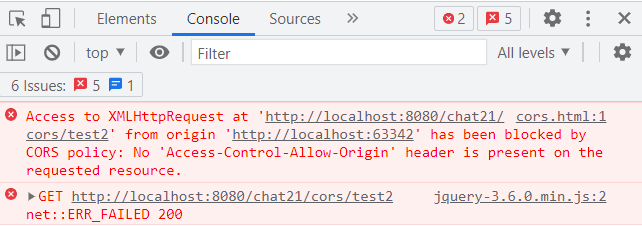
今天又给大家带来了一个很重要的知识点：SpringMVC中如何处理跨域问题，本文的内容同样适合于SpringBoot

**1、跨域访问报错**

当一个请求url的协议、域名、端口三者之间任意一个与当前页面url不同即为跨域。

出于安全原因，浏览器禁止Ajax调用驻留在当前原点之外的资源，比如从a.com发送一个ajax请求到b.com，则浏览器控制台会报跨域访问错误。

如下图，从http://localhost:63342/站点页面中向ttp://localhost:8080/chat21/cors/test2发送一个ajax请求，则出现了红色的错误信息，错误中包含了Access-Controll-Allow-Origin这样字样的错误，以后看到这个的时候，大家就要一眼看出来这是跨域问题。



## 2、同源定义

同源策略是浏览器的一个重要的安全策略，它用于限制一个源的文档或其加载的脚本如何与另外一个源进行交互，它能够隔绝恶意文档，减少被攻击的媒介。

如果两个URL的**协议**、**主机名**和**端口号**都是相同的，那么这两个URL就是同源的，否则不同源，不同源的访问就会出现跨域问题，就会出现上面的错误。

下表给出了与 URL http://store.company.com/dir/page.html 的源进行对比的示例:

| **URL** | **结果** | **原因** |
| --- | --- | --- |
| http://store.company.com/dir2/other.html | 同源 | 只有路径不同 |
| https://store.company.com/secure.html | 非同源 | 协议不同 |
| http://store.company.com:81/dir/etc.html | 非同源 | 端口号不同 |
| http://news.company.com/dir/other.html | 非同源 | 主机名不同 |

也就是说当在http://store.company.com/dir/page.html这个网站中向https://store.company.com、http://store.company.com:81和http://news.company.com三个地址发起AXJX请求都会失败并且会报跨域的错误。这就是浏览器的同源策略，只能访问同源的数据。

## 3、跨域问题如何解决？

跨域问题需要使用CORS来解决，请求端和后端接口需要遵循CORS规则来通信，便可解决跨域访问的问题。

CORS全称Cross-Origin Resource Sharing, 即跨域资源共享，是一个由一系列HTTP头组成的系统，这些HTTP头决定浏览器是否阻止前端javascript代码获取跨域请求的响应。为什么需要CORS ？这是因为浏览器存在同源安全策略，当我们在当前域请求另外一个域的资源时，浏览器默认会阻止脚本读取它的响应，这时CORS就有了用武之地。

跨源资源共享（CORS）是由大多数浏览器实现的W3C规范，允许您灵活地指定什么样的跨域请求被授权，而不是使用一些不太安全和不太强大的策略，如IFRAME或JSONP等。

## 4、CORS原理

CORS的原理：简单点说，就是在请求头或响应头中添加了一些配置，通过这些配置来便可轻松解决跨域问题。

想详细了解CORS原理的，建议先阅读下面2篇文章，然后再继续向下看，否则，最后你知道SpringMVC是如何解决的，但是不知道本质的原理是什么。

* CORS通信：http://itsoku.com/article/197
* 浏览器安全策略 & CORS：http://itsoku.com/article/198

## 5、SpringMVC中如何解决跨域问题？

SpringMVC内部提供了跨域问题的解决方案，只需要做一些简单的配置，而接口基本上不用做任何修改，便可解决跨域问题。

SpringMVC解决跨域问题的原理也就是SpringMVC遵循了CORS通信的规则来解决了跨域的问题，在响应头中添加了一些CORS需要的信息。

SpringMVC中提供了3种方案来解决跨域问题，下面一起来了解下。

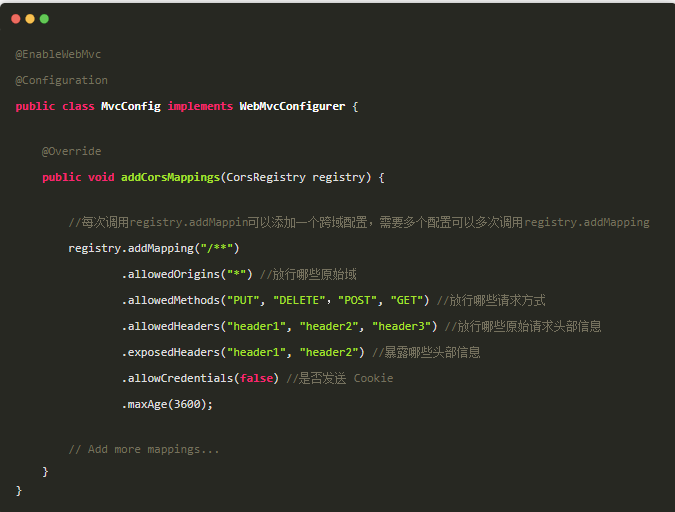
**6、方案1：方法或者类上标注@CrossOrigin注解**

* 接口方法上标注org.springframework.web.bind.annotation.CrossOrigin注解，如下test1接口上标注了@CrossOrigin注解，这个接口就支持跨域访问，@CrossOrigin注解中含有更详细的配置，这里就不细说了
* 也可以在类上标注@CrossOrigin注解，那么这个类中所有接口会支持跨域访问
* 也可同时在类和方法上标注@CrossOrigin注解，最后方法上的跨域访问会取合并后的配置



## 7、方案2：全局配置的方式

除了细粒度、基于注释的配置之外，您还可能需要定义一些全局CORS配置，这类似于使用筛选器，但可以声明为Spring MVC并结合细粒度@CrossOrigin配置。默认情况下，所有origins and GET, HEAD and POST methods是允许的。



## 8、方案3：拦截器的方式CorsFilter

