前后端分离项目，如何解决跨域问题

原创 梦想de星空 macrozheng 2019-07-31 08:33

收录于合集
#mall学习教程（技术要点篇）
17个

> 跨域资源共享（CORS）是前后端分离项目很常见的问题，本文主要介绍当SpringBoot应用整合SpringSecurity以后如何解决该问题。

什么是跨域问题

CORS全称Cross-Origin Resource Sharing，意为跨域资源共享。当一个资源去访问另一个不同域名或者同域名不同端口的资源时，就会发出跨域请求。如果此时另一个资源不允许其进行跨域资源访问，那么访问的那个资源就会遇到跨域问题。

跨域问题演示及解决

> 我们使用mall项目的源代码来演示一下跨域问题。此时前端代码运行在8090端口上，后端代码运行在8080端口上，由于其域名都是localhost，但是前端和后端运行端口不一致，此时前端访问后端接口时，就会产生跨域问题。

点击前端登录按钮

> 此时发现调用登录接口时出现跨域问题。

覆盖默认的CorsFilter来解决该问题

> 添加GlobalCorsConfig配置文件来允许跨域访问。

```
package com.macro.mall.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.cors.CorsConfiguration;
import org.springframework.web.cors.UrlBasedCorsConfigurationSource;
```

```
import org.springframework.web.filter.CorsFilter;

/**
 * 全局跨域配置
 * Created by macro on 2019/7/27.
 */
@Configuration
public class GlobalCorsConfig {

    /**
     * 允许跨域调用的过滤器
     */
    @Bean
    public CorsFilter corsFilter() {
        CorsConfiguration config = new CorsConfiguration();
        //允许所有域名进行跨域调用
        config.addAllowedOrigin("*");
        //允许跨越发送cookie
        config.setAllowCredentials(true);
        //放行全部原始头信息
        config.addAllowedHeader("*");
        //允许所有请求方法跨域调用
        config.addAllowedMethod("*");
        UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();
        source.registerCorsConfiguration("/**", config);
        return new CorsFilter(source);
    }
}
```

重新运行代码，点击登录按钮

> 发现需要登录认证的/admin/info接口的OPTIONS请求无法通过认证，那是因为复杂的跨越请求需要先进行一次OPTIONS请求进行预检，我们的应用整合了SpringSecurity，对OPTIONS请求并没有放开登录认证。

设置SpringSecurity允许OPTIONS请求访问

> 在SecurityConfig类的configure(HttpSecurity httpSecurity)方法中添加如下代码。

```
.antMatchers(HttpMethod.OPTIONS)//跨域请求会先进行一次options请求
.permitAll()
```

重新运行代码，点击登录按钮

> 发现已经可以正常访问。

一次完整的跨域请求

先发起一次OPTIONS请求进行预检

- 请求头信息：

```
Access-Control-Request-Headers: content-type
Access-Control-Request-Method: POST
Origin: http://localhost:8090
Referer: http://localhost:8090/
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Ge
```

- 响应头信息：

```
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: content-type
Access-Control-Allow-Methods: POST
Access-Control-Allow-Origin: http://localhost:8090
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Content-Length: 0
Date: Sat, 27 Jul 2019 13:40:32 GMT
Expires: 0
Pragma: no-cache
Vary: Origin, Access-Control-Request-Method, Access-Control-Request-Headers
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
```

- 请求成功返回状态码为200

发起真实的跨域请求

- 请求头信息：

```
Accept: application/json, text/plain, */*
Content-Type: application/json;charset=UTF-8

Origin: http://localhost:8090
Referer: http://localhost:8090/
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Ge
{username: "admin", password: "123456"}
password: "123456"
username: "admin"
```

- 响应头信息：

```
Access-Control-Allow-Credentials: true
Access-Control-Allow-Origin: http://localhost:8090
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Content-Type: application/json;charset=UTF-8
Date: Sat, 27 Jul 2019 13:40:32 GMT
Expires: 0
Pragma: no-cache
Transfer-Encoding: chunked
Vary: Origin, Access-Control-Request-Method, Access-Control-Request-Headers
```

```
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
```

- 请求成功返回状态码为200

项目源码地址

https://github.com/macrozheng/mall

推荐阅读

- 如何写出优雅的开源项目文档
- 订单模块数据库表解析（三）
- 订单模块数据库表解析（二）
- 订单模块数据库表解析（一）
- 商品模块数据库表解析（二）
- 商品模块数据库表解析（一）
- mall数据库表结构概览

欢迎关注，点个在看

收录于合集 #mall学习教程（技术要点篇） 17

上一篇
SpringBoot应用中使用AOP记录接口访问日志

下一篇
Java 8都出那么久了，Stream API了解下?

阅读原文

喜欢此内容的人还喜欢

项目中到底该不该用Lombok?

macrozheng