跨域访问CORS

2021年9月27日 10:08

跨域的服务调用在今天很普遍，因为我们大多时候会把前端和后端分开
前端可能会有一台Web服务器来存放html、jsp代码、小程序等，后端则采用另一台服务器去运行Servlet容器及其中的Spring框架
由于前端和后端不是一个服务器，那么就会产生跨域服务调用的问题

在浏览器中，为了安全性，**一般是不允许跨域访问的**，比如是不允许A服务器上JavaScript代码向B服务器发出请求
但我们要尽量让这种跨域访问变成可能

跨域服务调用有两种：简单请求、非简单请求，因为我们传的数据是application/json格式，因此是非简单请求

非简单的跨域请求

浏览器在向服务器发出真正的请求之前，它先要做一次询问，称之为**预检请求（OPTION REQUEST）**，即对一个URL发出OPTION的操作，这个OPTION Request中包含了静态网页、html、jsp代码等来源的域名，以及要做什么操作
例：

```
OPTIONS /cors HTTP/1.1
Origin: http://api.bob.com
Access-Control-Request-Method: PUT
Access-Control-Request-Headers: X-Custom-Header
Host: api.alice.com
Accept-Language: en-US
Connection: keep-alive
User-Agent: Mozilla/5.0...
```

在这个例子中，来源Origin是http://api.bob.com，计划用PUT的方式去访问api.alice.com
当api.alice.com接收到这样的一个预检请求后，会检查请求中的Origin、访问方法、包头(Headers)是否在本服务器允许的跨域访问范围内，如果都在允许范围内，服务器给出HTTPResponse回应，否则会报错，不作出回应

服务器给出的HTTPResponse回应如下：

```
HTTP/1.1 200 OK
Date: Mon, 01 Dec 2008 01:15:39 GMT
Server: Apache/2.0.61 (Unix)
Access-Control-Allow-Origin: http://api.bob.com
Access-Control-Allow-Methods: GET, POST, PUT
Access-Control-Allow-Headers: X-Custom-Header
Content-Type: text/html; charset=utf-8
Content-Encoding: gzip
Content-Length: 0
Keep-Alive: timeout=2, max=100
Connection: Keep-Alive
Content-Type: text/plain
```

在回应中，可以看到允许的来源、方法、包头等

之后，客户端就可以向服务器发出正式的跨域请求了，即发出CORS Request
CORS请求如下：

```
PUT /cors HTTP/1.1
Origin: http://api.bob.com
Host: api.alice.com
X-Custom-Header: value
Accept-Language: en-US
Connection: keep-alive
User-Agent: Mozilla/5.0...
```

在每一个CORS请求中，都会包含一个Origin的包头，来标识请求的来源，通过检查这个来源，服务器来决定是否能对这个请求作出回应

SpringMVC中具体配置服务器的允许范围

通过WebMVCConfigurer接口的addCORSMapping方法来配置

有多种方法可以进行配置跨域访问

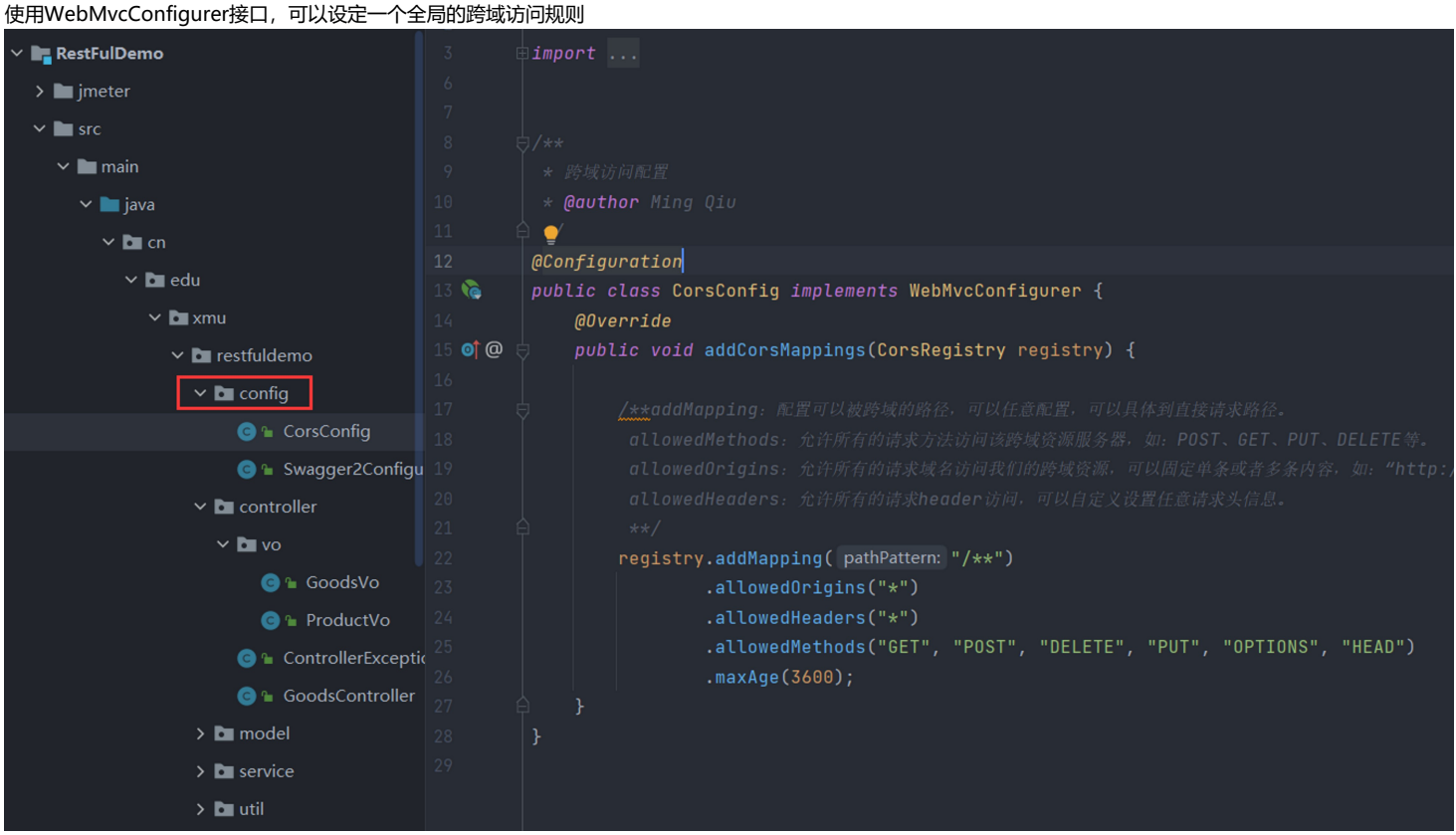
以Restfuldemo为例：

用注解@CrossOrigin配置：

这个注解可以在类前面、方法前面去设置

对服务器上所有请求设定统一来源：

使用WebMvcConfigurer接口，可以设定一个全局的跨域访问规则



一般我们所有的配置都放在config包下

在类前面有一个注解**@Configuration**，在之前的@SpringBootApplication有提及，这个注解表示这个类可以被Spring容器认为去配置信息，实现配置文件的功能

这个接口中有很多方法，我们通过实现其中的addCORSMappings来设定跨域访问的配置

参数是一个CorsRegistry类型的，通过这个参数把跨域设置到系统中去

registry的几个方法如下：

addMapping：配置可以被跨域的路径，可以任意配置，可以具体到直接请求路径。

allowedMethods：允许所有的请求方法访问该跨域资源服务器，如：POST、GET、PUT、DELETE等。

allowedOrigins：允许所有的请求域名访问我们的跨域资源，可以固定单条或者多条内容，如：“http://www.aaa.com”，只有该域名可以访问我们的跨域资源。

allowedHeaders：允许所有的请求header访问，可以自定义设置任意请求头信息。

maxAge：有效时间