链滴

项目支持跨域访问 -java

作者：judge

原文链接：https://ld246.com/article/1597033937031

来源网站：链滴

前后端分离项目

SpringBoot - vue 3.0

跨域设置

java后台配置跨域

```
package com.xd.pre.common.config;

import org.apache.commons.lang3.StringUtils;
import org.springframework.context.annotation.Configuration;

import javax.servlet.*;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 * @author Judge
 * @date 2020/8/10 15:45
 */
@Configuration
public class CorsFilter implements Filter {

    @Override
    public void doFilter(ServletRequest req, ServletResponse res, FilterChain chain) throws IOEx
ception, ServletException {
        HttpServletResponse response = (HttpServletResponse) res;
        HttpServletRequest request = (HttpServletRequest) req;

        // 不使用*，自动适配跨域域名，避免携带Cookie时失效
        String origin = request.getHeader("Origin");
        if(StringUtils.isNotBlank(origin)) {
            response.setHeader("Access-Control-Allow-Origin", origin);
        }

        // 自适应所有自定义头
        String headers = request.getHeader("Access-Control-Request-Headers");
        if(StringUtils.isNotBlank(headers)) {
            response.setHeader("Access-Control-Allow-Headers", headers);
            response.setHeader("Access-Control-Expose-Headers", headers);
        }

        // 允许跨域的请求方法类型
        response.setHeader("Access-Control-Allow-Methods", "*");
        // 预检命令（OPTIONS）缓存时间，单位：秒
        response.setHeader("Access-Control-Max-Age", "3600");
        // 明确许可客户端发送Cookie，不允许删除字段即可
        response.setHeader("Access-Control-Allow-Credentials", "true");

        chain.doFilter(request, response);
    }
```

```
    @Override
    public void init(FilterConfig filterConfig) {

    }

    @Override
    public void destroy() {
    }


}
```

2.VUE 本地开发环境配置

```
module.exports = {
  dev: {
    host: '127.0.0.1',
    port: 8084,
    open: true,// vue项目启动时自动打开浏览器
    proxy: {
      '/api': { // '/api'是代理标识，用于告诉node，url前面是/api的就是使用代理的
        target: "http://localhost:8080", //目标地址，一般是指后台服务器地址
        changeOrigin: true, //是否跨域
        pathRewrite: { // pathRewrite 的作用是把实际Request Url中的'/api'用""代替
          '^/api': ""
        }
      }
    }
  }
}
```

配置完成后，我们发起的每次http请求的Request Url的前面一部分都会和我们本地的源一样；

举例：

```
Axios.get("/api/XX")
```

我们项目跑在开发环境下的http://localhost:8080上，那么上述请求的Request Url的就是 http://localhost:8080/api/XX;这应该就是我们的代理服务器中该资源的地址，与客户端同源；而数据的实际来则是http://localhost:8080/api/XX，这样即实现代理跨域；

Nginx代理——生产环境

nginx版本：1.16.1

修改nginx.conf文件下的serve下配置：

**

```
        listen       80;  //端口号
        server_name  localhost;
        // 打包之后项目放到的路径；
        location / {
```

```
    root   dist;  // dist是指和nginx.exe同级的dist目录，一般默认是html文件夹
    index  index.html index.htm;  // dist文件夹下的html文件
}
// 此处是服务器代理部分，为解决http请求跨域问题
// 若服务器已设置允许跨域，则这块就不需要了
location  ^~api{
    proxy_pass http://127.0.0.1:8080/api;
        // 被代理服务器地址
}
```