Webpack跨域原理：

前言：

1. 跨域是由于ajax请求的同源策略导致的，也就是说，除了ajax请求，其他请求不存在跨域的说法
2. 服务器与服务器之间的http请求不存在跨域

Webpack解决跨域：

正因为上面这两条，所以webpack跨域的原理：

1. 在本地项目集成一个express框架（Nodejs的服务端框架，可以用来写后端，也可以用来写接口，也是一个MVC框架）
2. 遵循ajax的同源策略，本地请求本地，自然是同源，所以不会出现跨域
3. 根据前言中的第2条：服务器之前请求不存在跨域，所以express服务去请求后端服务器，自然也就不存在跨域
4. 最后Webpack把从后端服务器请求的数据返回给前端。

关于Webpack配置：

我们都知道 Vue-Cli3+ 版本的 devServer 是写在 vue.config.js 文件中的

1. 假如我们后端的域名是：https://www.xxx.com
2. 假如我们后端给我们提供了一个接口，获取一个列表，这个接口地址是：https://www.xxx.com/getList
3. 那么我们现在要通过配置去实现：

```
module.exports = {
    host: '127.0.0.1', // 自己本地项目运行在什么ip上，比如：localhost、127.0.0.1、0.0.0.0
    port: 8080, // 本地项目运行在哪个端口
    devServer: {
        proxy: {
            '/api/': {
                target: 'https://www.xx.com', // 这里写后端的域名或者ip + 端口
                changeOrigin: true, // 这个字段我得再研究一下
                secure: false, // 这个字段默认是true，表示不支持https协议，如果target是https协议，这个字段需要改成false
                pathRewrite: { // 这个字段表示将你的请求的url中的某一段替换成指定的字符串
                    '^/api': '' // 比如这里就表示将你请求的url中的api替换成空字符串
                    // 所以我们在请求接口的时候，路径一般都是api开头，但是我们的后端接口中并没有/api/这一层，所以这个时候pathRewrite就派上用场了，那么反过来，如果后端的接口是这样的：https://www.xxx.com/api/getList，这个时候，我们就不需要替换api了
                }
            }
        }
    }
```

```
}
```