# Angular+SpringBoot的http通信

# Angular设置代理

# 项目背景

1. SpringBoot 2.2.4
2. Angular 7.+

# 实现思路

因为前后端分离的开发，导致前端的请求存在跨域的问题。

对此，可以在Angular前端进行代理设置，解决跨域请求的问题。

# Angular具体代理配置步骤

## 配置基础api路径

|  |
| --- |
| **在environment.ts文件中添加一个属性，apiPath** |
| export const environment = {  production: false,  apiPath : '/api/mysql' // 这个的定义了请求时候的基本的路径  }; |

## 开发自定义的http请求服务

|  |
| --- |
| **【注意】开发前需要在app.module.ts文件中引入HttpClientModule模块** |
| **Ng g service common/api** |
| import { Injectable } from '@angular/core';  import { HttpClient } from '@angular/common/http';  import { Observable, from } from 'rxjs';  import { environment } from '../../environments/environment';  @Injectable({  providedIn: 'root'  })  export class ApiService {  //0.基础的请求前缀配置在environment文件中  private apiUrl = environment.apiPath;  //1.注入httpClient对象  constructor(private httpClient:HttpClient) { }  //2.create get method with httpClient Object  get(url:string,option?):Observable<any>{  //2.1 note:the url param should concat with apiUrl!This is very important  return this.httpClient.get(this.apiUrl+url,option);  }  } |

## 编写代理文件proxy-config.json

|  |
| --- |
| **【此文件直接在webapp这个根目录下面编写即可，名字随意格式json】** |
| {  "/api/\*":{ // 这个是遇到/api就开始解析操作  "target":"http://localhost:8080", // 这个就是跨域的地址和端口号  "secure":false,  "logLevel":"debug",  "changeOrigin":true    }  } |

## 配置代理文件到angular.json文件中

|  |
| --- |
| **需要找到该文件中的server的内容，并添加如下** |
| **"proxyConfig": "proxy-config.json" ，就是添加这一句即可** |
| "serve": {  "builder": "@angular-devkit/build-angular:dev-server",  "options": {  "browserTarget": "webapp:build",  "proxyConfig": "proxy-config.json"  },  "configurations": {  "production": {  "browserTarget": "webapp:build:production"  }  }  }, |

## 配置package.json文件中的启动命令（可选操作）

|  |
| --- |
| **【注意，此步骤为可选操作，即不设置也可以成功代理】** |
| **同样是在启动的时候指定代理文件** |
| "scripts": {  "ng": "ng",  "start": "ng serve --proxy-config proxy-config.json",  "build": "ng build",  "test": "ng test",  "lint": "ng lint",  "e2e": "ng e2e"  }, |

## 在模块中使用自定义的http请求，自动解析为代理的路径

|  |
| --- |
| const url = '/queryAllUsers\_m'; // 从这里开始就是正常的操作了！  this.apiService.get(url).subscribe(  (res)=>{  console.log(res);  }  ); |

# SpringBoot服务端开发

服务端方面，就是正常开发，controller层，service层，mapper，model, xml映射文件

# 总结

**Angular配置代理的操作还是比较简单的！**