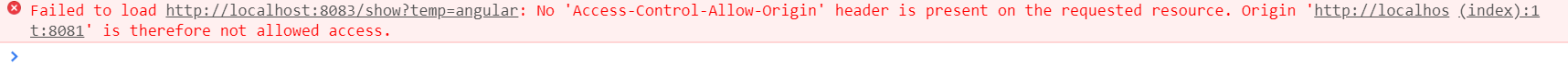
## 概念

跨域：浏览器对于javascript的同源策略的限制，例如a.cn下面的js不能调用b.cn的js,对象或数据（a.cn和b.cn是不同的域）；

跨域维度：协议、主机地址（ip、域名）、端口号。如果有一个不同，则表明进行了跨域。

例如：<http://localhost:8081> 中访问[http://localhost:8083/show?temp=”java](http://localhost:8083/show?temp=)” 就出现了跨域。



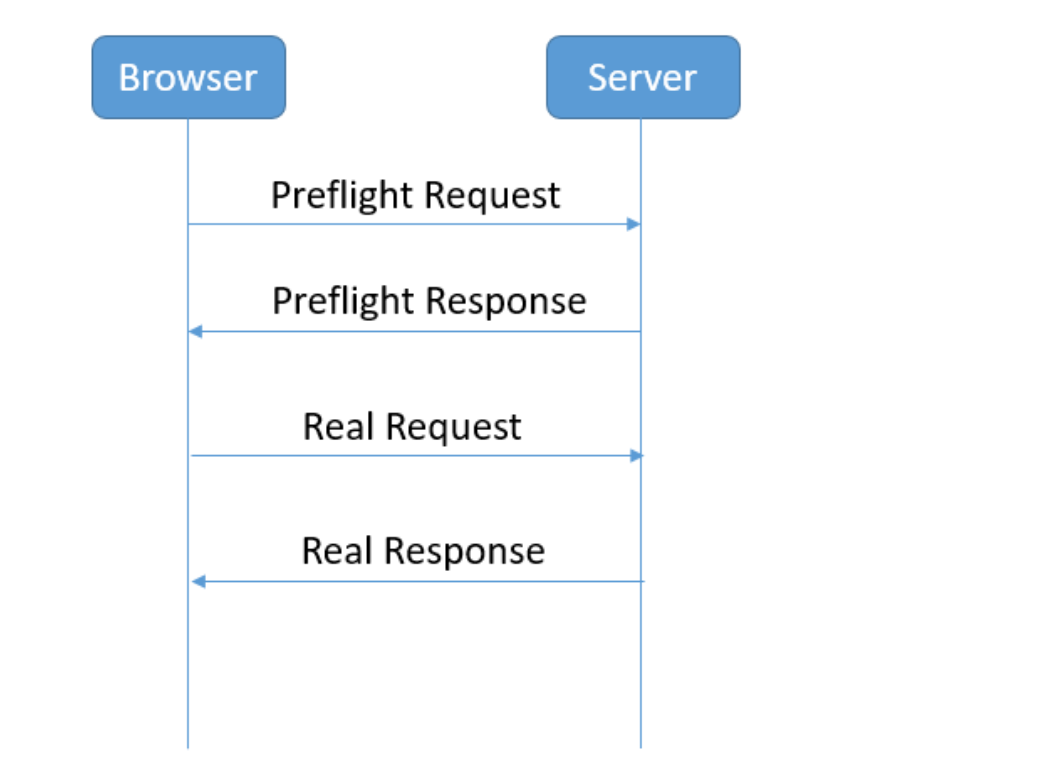
## 解决方法

### 跨域解决方案 CORS

CORS 是一个 W3C 标准，全称是"跨域资源共享"（Cross-origin resource sharing）。 CORS需要浏览器和服务器同时支持。目前，所有浏览器都支持该功能， IE 浏览器不能低于 IE10。

它允许浏览器向跨源服务器，发出 XMLHttpRequest 请求，从而克服了 AJAX 只能同源使用的限制。整个 CORS 通信过程，都是浏览器自动完成，不需要用户参与。对于开发者来说， CORS 通信与同源的 AJAX 通信没有差别，代码完全一样。浏览器一旦发现 AJAX 请求跨源，就会自动添加一些附加的头信息，有时还会多出一次附加的请求，但用户不会有感觉。因此，实现 CORS 通信的关键是服务器。只要服务器实现了 CORS 接口，就可以跨源通信。

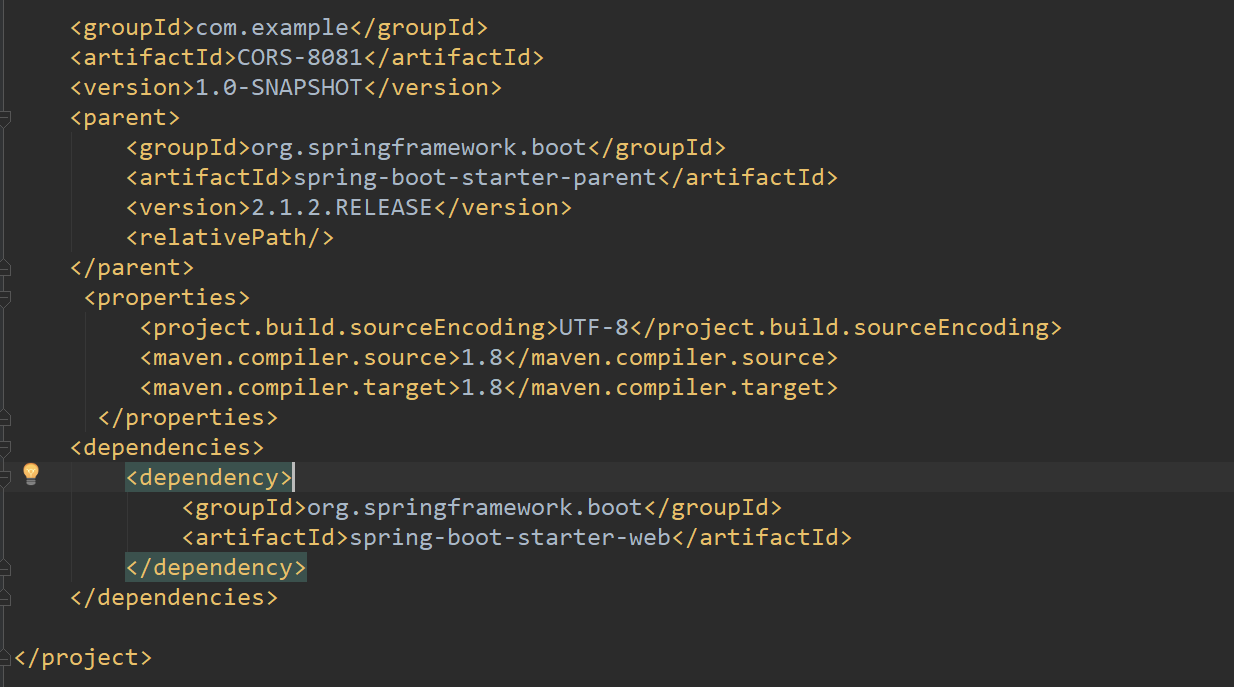
请求过程如下图:





案例：

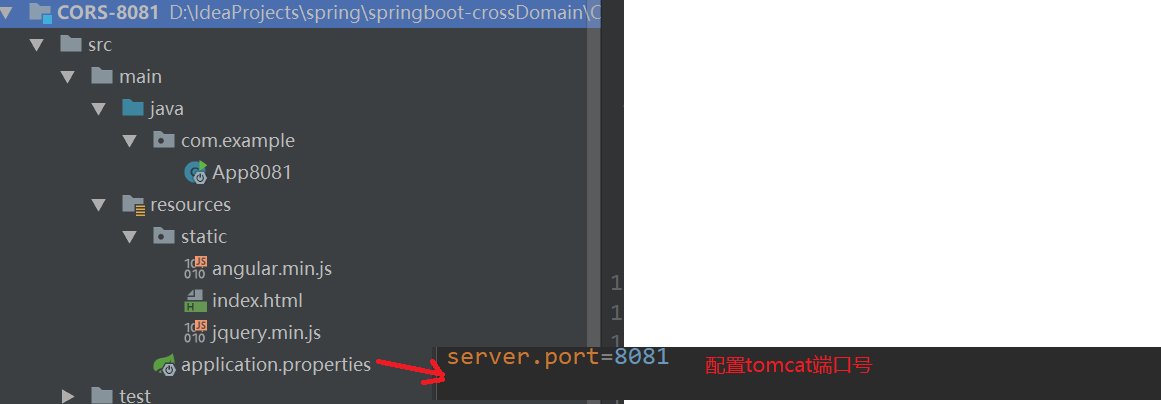
1. 创建maven项目CORS-8081和CORS-8082（pom.xml）



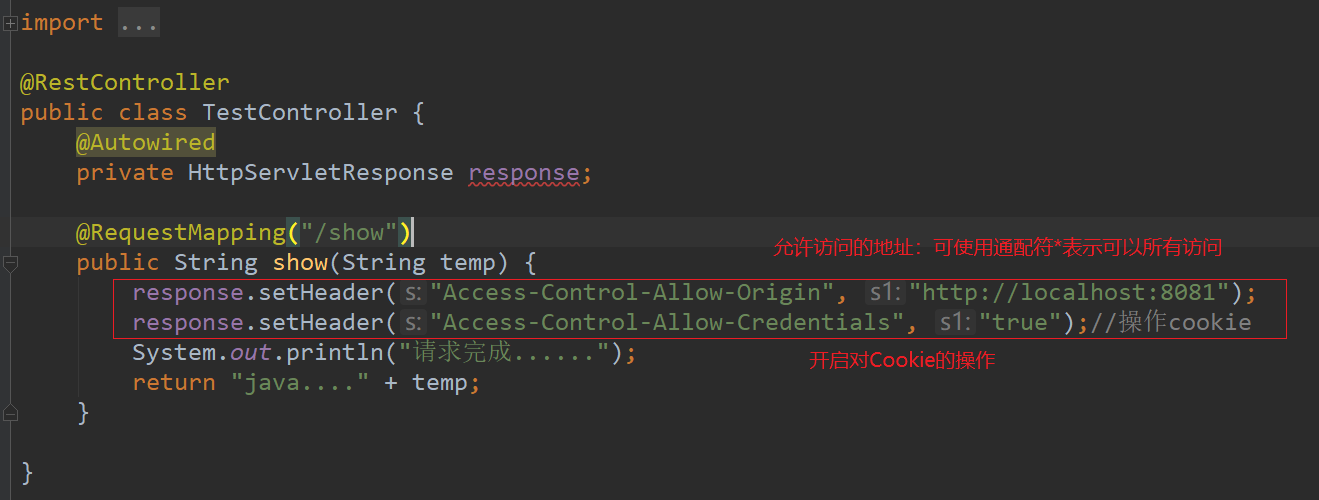
1. CORS-8081的resources下创建static目录，编写index页面



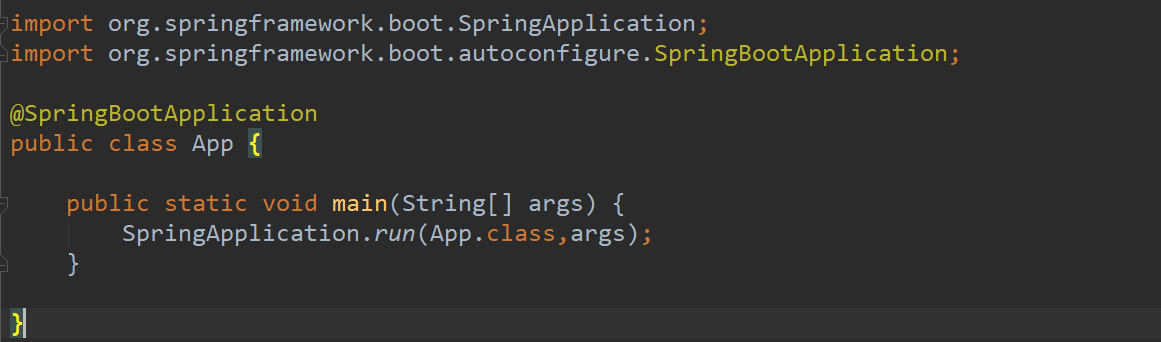
CORS-8081的目录结构



1. 修改CORS-8083的端口号为8083，添加controller方法



1. 启动类（两个一样）



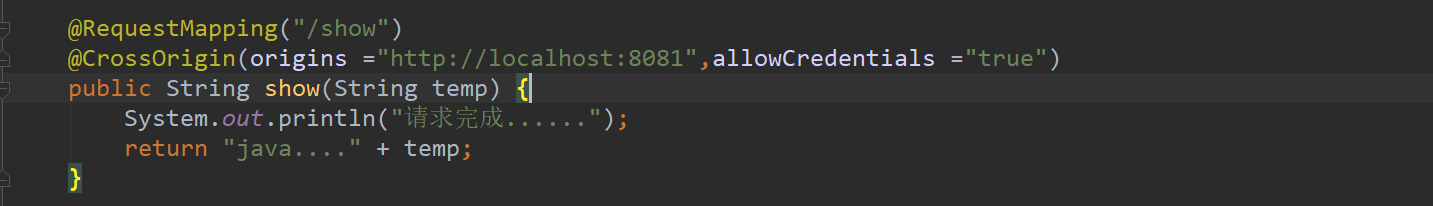
1. 测试



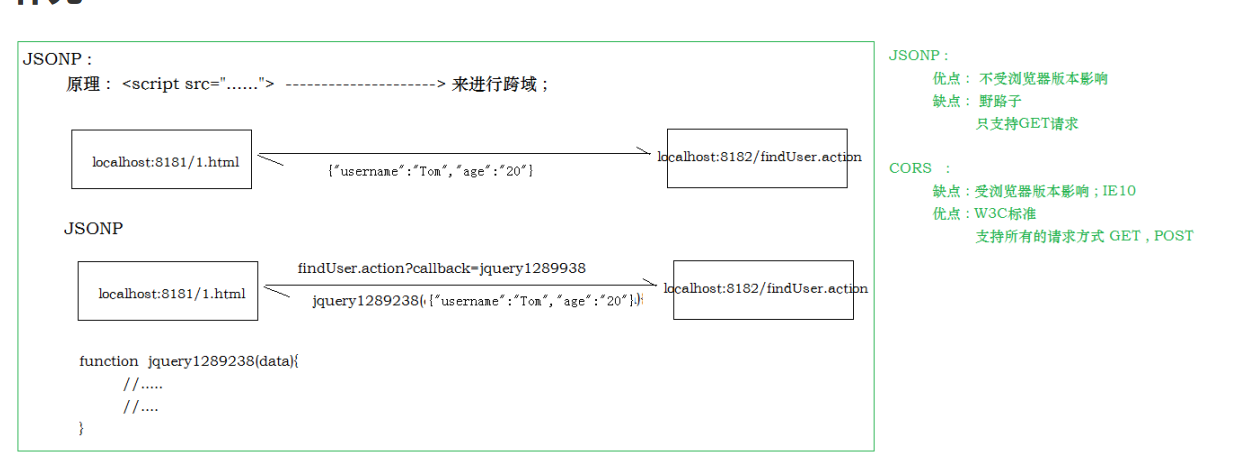
tips:

springMVC 的版本在 4.2 或以上版本，可以使用注解实现跨域, 我们只需要在需要跨域的方法上添加注解@CrossOrigin 即可@CrossOrigin(origins="http://localhost:9105",allowCredentials="true") allowCredentials="true" 可以缺省

使用注解配置，只需要在需要跨域的方法上添加注解@CrossOrigin即可

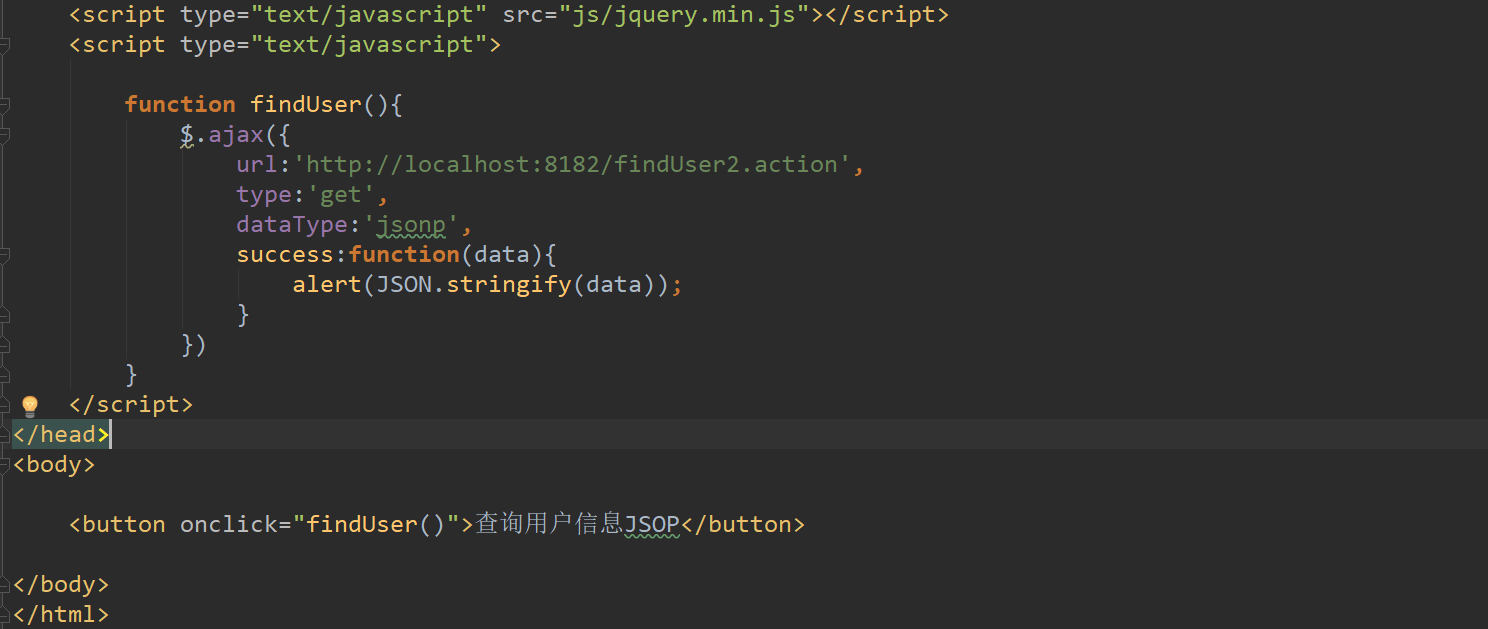


### JSONP



底层是通过script的src属性进行包装成方法，实现跨域

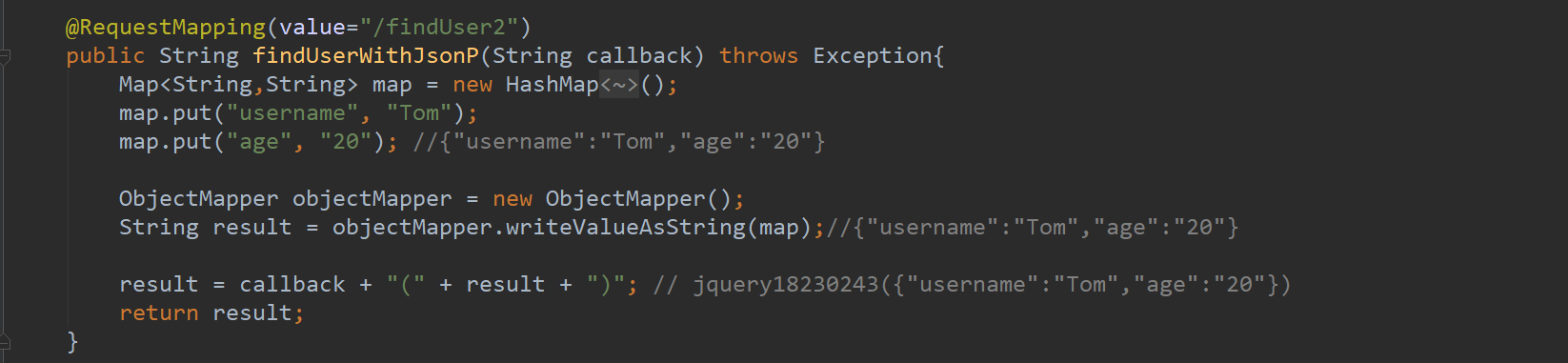
1. 使用jquery请求



1. 使用angularjs请求



controller处理



注：在springboot2.x以上，jsonp访问会出现错误