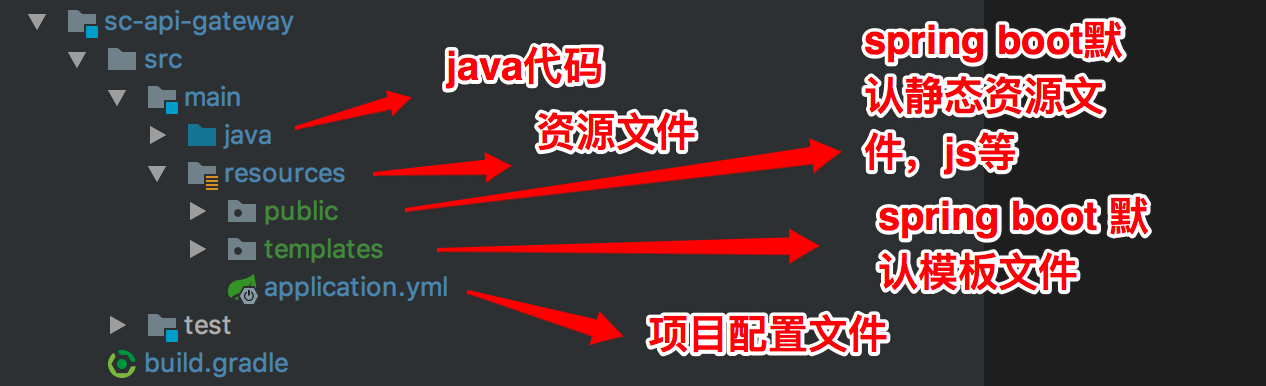
# 项目构建说明

该项目使用gradle作为构建工具，gradle安装说明和工具参见“14.开发工具文件夹”。文件夹结构和maven一致，根目录下的build.gradle为所有项目的公共配置，每个项目下的build.gradle为该项目构建文件，包括该项目的依赖等，项目文件夹说明如下图所示。



安装gradle后使用开发工具导入项目sc根文件夹，导入类型选择gradle

# 项目工具说明：

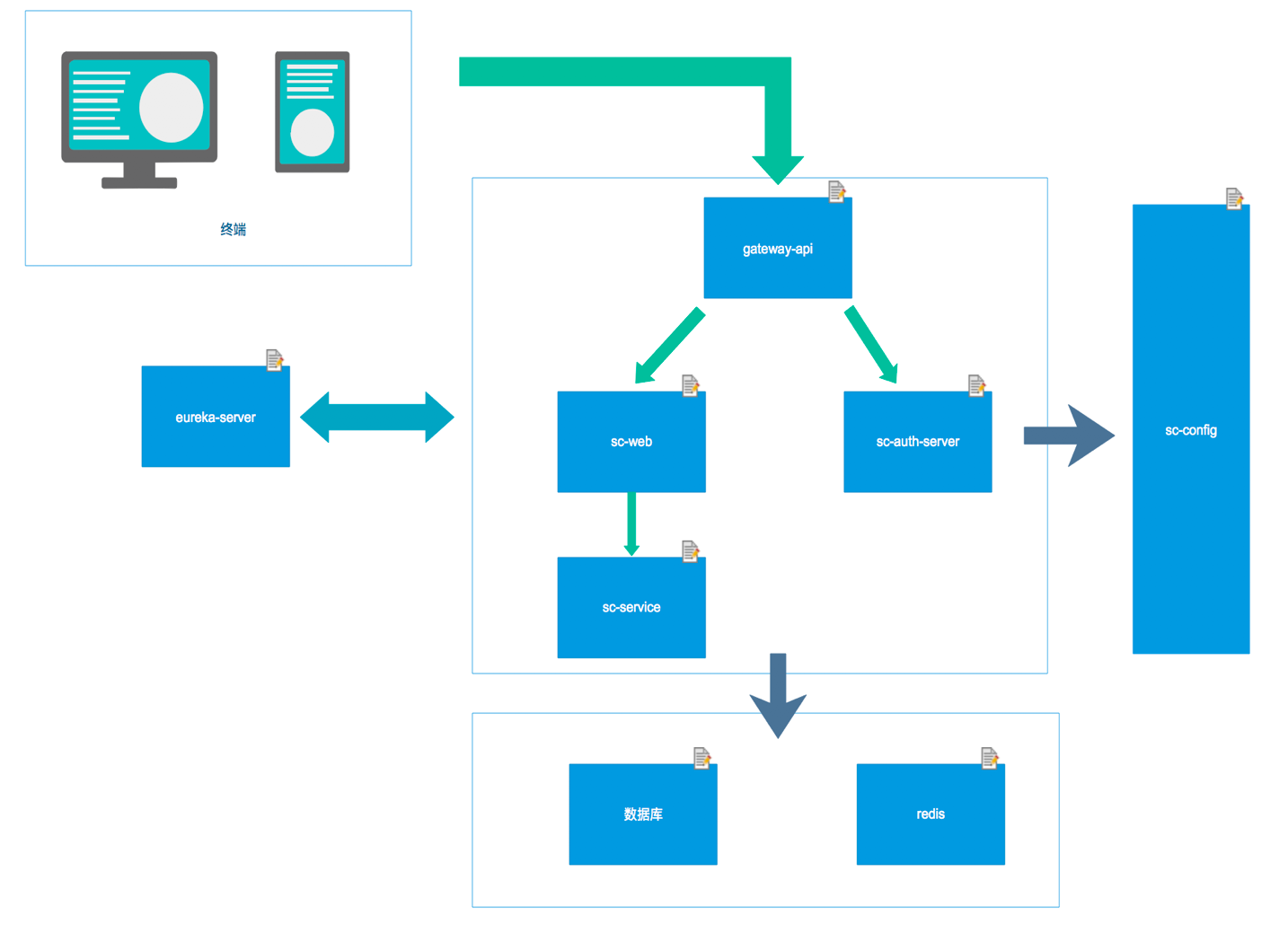
JDK:1.8

开发工具：eclipse、idea，都建议最新版，spring boot和gradle老版本支持不太，建议使用idea。

Mysql：5.7

Redis：3.x

# 系统架构图



# 项目启动顺序说明：

1. sc-eureka-server

服务注册中心，优先启动

1. sc-config-server

系统配置中心，各项目的配置文件都在该项目进行管理

1. sc-oauth-server

权限验证中心，需要redis和mysql

1. sc-api-gateway

系统网关, 进行负载均衡，所有的请求先通过网关，然后转发到其他的服务，需要redis

1. sc-web-auth

web登录及权限相关服务，需要mysql

1. sc-web

web相关入口项目，需要redis

1. sc-service

web服务提供项目，需要mysql

1. sc-admin-server

系统监控项目，不是必须启动

# 项目启动方法

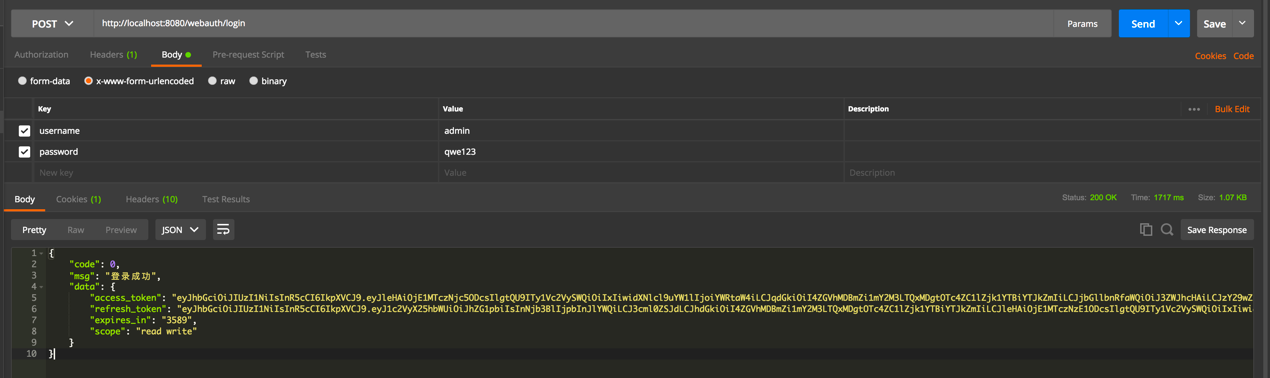
Spring boot 内嵌tomcat容器，所以不需要安装tomcat，使用main方法启动，每个项目中都有一个启动的main方法，以sc-eureka-server举例，在sc-eureka-server/src/main/java/app文件夹中的ScEurekaApplication为项目启动类，里面含有一个main的启动方法，直接按照执行普通类main方法即可。



# 项目权限说明

项目登陆后会获取一个token，前端需要缓存该参数，以后的每次请求都需要携带该token进行访问，如果token过期或token非法会拒绝访问。

## 登录获取token的请求如下



请求：

url: <http://localhost:8080/webauth/login>

method：post

content-type：x-www-form-urlencoded

参数需要两个：username：admin

Password：qwe123

返回：

{

"code": 0,

"msg": "登录成功",

"data": {

"access\_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTczNjc5ODcsIlgtQU9ITy1Vc2VySWQiOiIxIiwidXNlcl9uYW1lIjoiYWRtaW4iLCJqdGkiOiI4ZGVhMDBmZi1mY2M3LTQxMDgtOTc4ZC1lZjk1YTBiYTJkZmIiLCJjbGllbnRfaWQiOiJ3ZWJhcHAiLCJzY29wZSI6WyJyZWFkIiwid3JpdGUiXX0.k9ZBBPDDw41kNyNAJKAvIRvWCag4F53Arr4A2SR-m8g",

"refresh\_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyX25hbWUiOiJhZG1pbiIsInNjb3BlIjpbInJlYWQiLCJ3cml0ZSJdLCJhdGkiOiI4ZGVhMDBmZi1mY2M3LTQxMDgtOTc4ZC1lZjk1YTBiYTJkZmIiLCJleHAiOjE1MTczNzE1ODcsIlgtQU9ITy1Vc2VySWQiOiIxIiwianRpIjoiMDRjM2Q1ZGMtODZiZS00Y2ZjLThjNzYtYWIxZjcwOGI3MDA5IiwiY2xpZW50X2lkIjoid2ViYXBwIn0.qlyM4hmRuknrDl6EJBGdbfQQ0IGGhVRXAW23vENpCcI",

"expires\_in": "3589",

"scope": "read write"

}

}

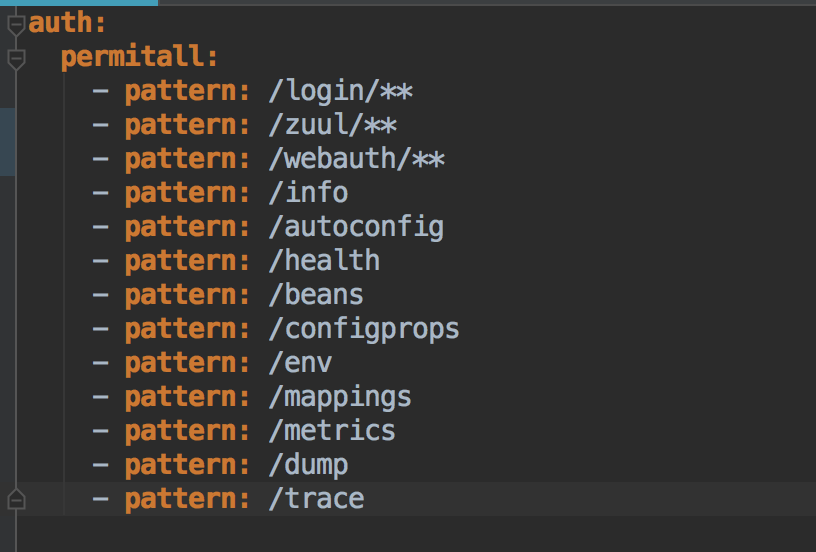
## 业务请求：

每次的请求在url附带access\_token=xxxxxx参数，样例如下

[*http://localhost:8080/web/hello1?access\_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTczNjc5ODcsIlgtQU9ITy1Vc2VySWQiOiIxIiwidXNlcl9uYW1lIjoiYWRtaW4iLCJqdGkiOiI4ZGVhMDBmZi1mY2M3LTQxMDgtOTc4ZC1lZjk1YTBiYTJkZmIiLCJjbGllbnRfaWQiOiJ3ZWJhcHAiLCJzY29wZSI6WyJyZWFkIiwid3JpdGUiXX0.k9ZBBPDDw41kNyNAJKAvIRvWCag4F53Arr4A2SR-m8g*](http://localhost:8080/web/hello1?access_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTczNjc5ODcsIlgtQU9ITy1Vc2VySWQiOiIxIiwidXNlcl9uYW1lIjoiYWRtaW4iLCJqdGkiOiI4ZGVhMDBmZi1mY2M3LTQxMDgtOTc4ZC1lZjk1YTBiYTJkZmIiLCJjbGllbnRfaWQiOiJ3ZWJhcHAiLCJzY29wZSI6WyJyZWFkIiwid3JpdGUiXX0.k9ZBBPDDw41kNyNAJKAvIRvWCag4F53Arr4A2SR-m8g)

## 不进行权限拦截

如果不需要进行权限拦截，需要在sc-api-gateway项目中的application.yml文件中进行配置。



# 服务间相互调用

Spring cloud服务间的调用使用http协议，所以方便进行测试，测试工具可以使用postman。

1. 定义服务间调用的接口

参见sc-service-api项目中的com.neusoft.service.api.helloservice.service. HelloServiceRemoteApi，其中定义了3个方法，说明了3中服务接口传值的方式

1. 在被调用端实现该接口

参见sc-service项目中的com.neusoft.service.hello.service.HelloServiceRemoteApiImpl

1. 使用注解在调用端定义远程接口

参见com.neusoft.web.hello.service.HelloBackgroundService，

@FeignClient(value = "sc-service", fallback = HelloBackgroundFallbackService.class)

其中sc-service为被调用端的应用名，fallback为调用失败后的处理方法，可以省略。

1. 在调用端注入该接口

参见sc-service项目中的com.neusoft.web.hello.controller.HelloController