目录

[一、 模块和配置 1](#_Toc4885896)

[1. 注册中心 2](#_Toc4885897)

[1) bootstrap.yml 2](#_Toc4885898)

[2) 启动注册中心 2](#_Toc4885899)

[2. 配置中心 4](#_Toc4885900)

[1) bootstrap.yml 5](#_Toc4885901)

[2) {profile} 5](#_Toc4885902)

[3) 端口 7](#_Toc4885903)

[4) 注册到注册中心 7](#_Toc4885904)

[5) 注册中心里的显示 8](#_Toc4885905)

[3. 用户中心 9](#_Toc4885906)

[1) 数据库脚本 9](#_Toc4885907)

[2) bootstrap.yml 10](#_Toc4885908)

[3) bear-user-center.yml 10](#_Toc4885909)

[a) 日志级别和文件配置 11](#_Toc4885910)

[b) 数据源配置 11](#_Toc4885911)

[c) Rabbitmq配置 11](#_Toc4885912)

[d) Mybatis配置 12](#_Toc4885913)

[e) 微信公众号配置 12](#_Toc4885914)

[4) 配置类 13](#_Toc4885915)

[a) 全局异常处理 13](#_Toc4885916)

[b) Rabbitmq的exchange声明 14](#_Toc4885917)

[c) 资源权限配置 14](#_Toc4885918)

[d) 密码加密处理器 15](#_Toc4885919)

[4. 认证中心 15](#_Toc4885920)

[1) 数据库脚本 16](#_Toc4885921)

[2) bootstrap.yml 16](#_Toc4885922)

[3) bear-auther-center.yml 16](#_Toc4885923)

[a) redis配置 16](#_Toc4885924)

[b) 数据库配置 17](#_Toc4885925)

[c) token是否用jwt 17](#_Toc4885926)

[4) 配置类 19](#_Toc4885927)

[a) 授权服务器配置 20](#_Toc4885928)

[b) 资源服务器 20](#_Toc4885929)

[c) Session共享配置 20](#_Toc4885930)

[5. 文件中心 21](#_Toc4885931)

[1) bear-file-center.yml 22](#_Toc4885932)

[a) 数据库和mq 22](#_Toc4885933)

[b) 上传文件大小限制 22](#_Toc4885934)

[c) 自定义配置-本地存储文件 22](#_Toc4885935)

[d) 阿里云存储文件 23](#_Toc4885936)

[2) 配置类 23](#_Toc4885937)

[a) 加载jar包外文件 23](#_Toc4885938)

[b) 资源服务器 24](#_Toc4885939)

[6. 网关 25](#_Toc4885940)

[1) Bear-gateway-zuul.yml 26](#_Toc4885941)

[a) 路由规则 26](#_Toc4885942)

[b) 自定义参数 26](#_Toc4885943)

[2) 配置类 27](#_Toc4885944)

[a) 跨域配置 28](#_Toc4885945)

[b) 异常处理 28](#_Toc4885946)

[7. 日志组件log-starter 29](#_Toc4885947)

[1) spring.factories 29](#_Toc4885948)

[2) 使用该组件 30](#_Toc4885949)

[8. 日志中心 31](#_Toc4885950)

[1) bear-log-center.yml 31](#_Toc4885951)

[2) 配置类 32](#_Toc4885952)

[a) 开启异步线程池 33](#_Toc4885953)

[b) 声明队列 33](#_Toc4885954)

[3) 处理日志消息 34](#_Toc4885955)

[4) 日志存储mysql和elasticsearch切换 34](#_Toc4885956)

[9. 后台管理系统 35](#_Toc4885957)

[1) Bear-system-admin.yml 36](#_Toc4885958)

[a) 数据库和mq 36](#_Toc4885959)

[b) 邮件配置 36](#_Toc4885960)

[2) 消息处理 37](#_Toc4885961)

[3) 静态资源 38](#_Toc4885962)

[10. 监控中心 39](#_Toc4885963)

[11. 通知中心 40](#_Toc4885964)

[1) bear-notification-center.yml 40](#_Toc4885965)

[12. Swagger配置 41](#_Toc4885966)

[13. user-info-uri 41](#_Toc4885967)

[14. bear-common模块 42](#_Toc4885968)

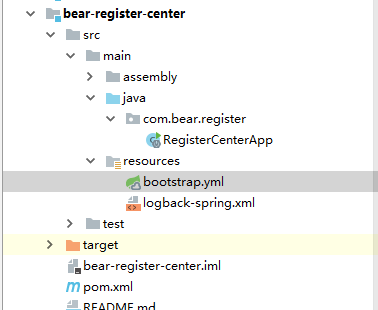
[1) PermitAllUrl 42](#_Toc4885969)

[2) AppUserUtil 43](#_Toc4885970)

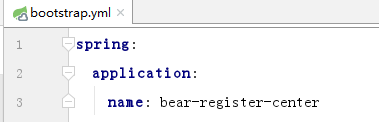
# 模块和配置

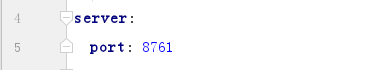
|  |  |
| --- | --- |
| **工程名** | **说明** |
| bear-common | 工具类、公共常量等 |
| bear-config-center | 配置中心 |
| bear-file-center | 文件中心 |
| bear-gateway-zuul | 网关 |
| bear-log-center | 日志中心 |
| bear-log-starter | 日志组件，别的项目直接引入该依赖即可 |
| bear-system-admin | 后台管理 |
| bear-monitor-center | 监控中心 |
| bear-auth-center | 认证中心 |
| bear-register-center | 注册中心 |
| bear-user-center | 用户中心、用户、角色、权限 |

## 注册中心



### bootstrap.yml

给该应用起个名字

应用启动后的端口号

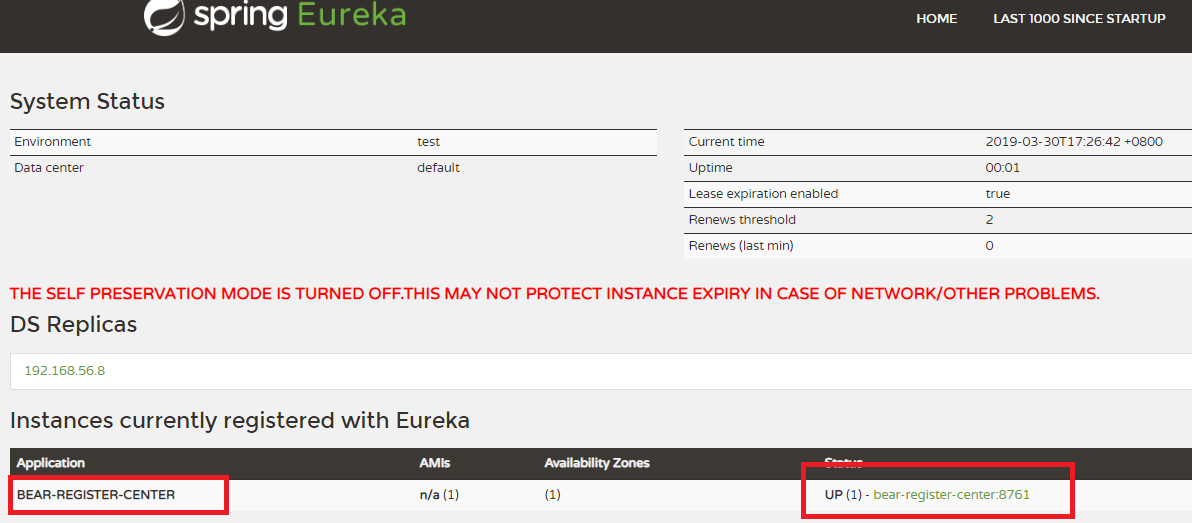
### 启动注册中心

直接运行下面类的main方法

**com.bear.register.RegisterCenterApp.java**

访问http://localhost:8761

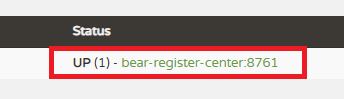
8761是bootstrap.yml里配置的系统端口号



Application列下的，BEAR-REGISTER-CENTER就是我们在bootstrap.yml里指定的应用名，注册中心给我们大写处理了。

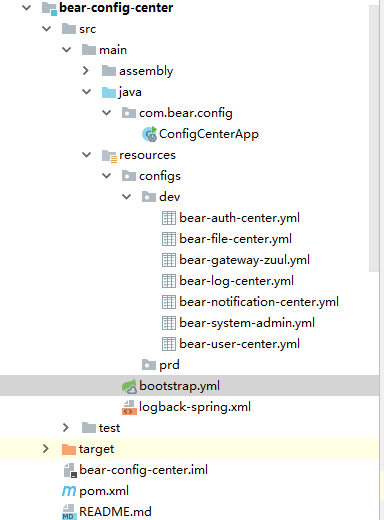
Status下显示的是我们在bootstrap.yml里的instance-id，如下图，我们这里是应用名加端口号

该配置为true的话，是用ip注册，否则是主机名注册，强烈建议配置为true

 http://192.168.56.8:8761/info点击这里会跳转到类似如下的地址

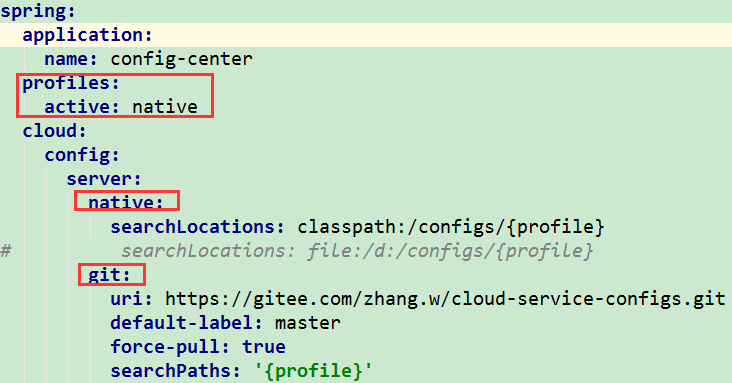
用这种方式，我们可以看到服务的具体ip地址和端口

## 配置中心



### bootstrap.yml

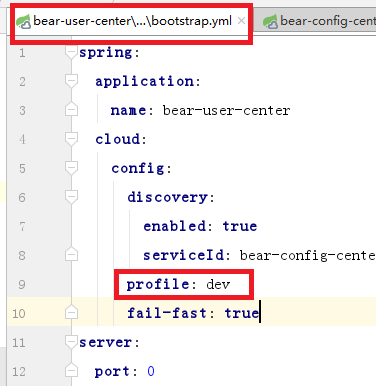
配置在本地或者git



通过spring.profiles.active这里可以指定配置文件在本地，还是classpath下还是在远程git上面，这里默认是放在了本地的classpath下，这里可根据实际项目需求和运维条件进行合理的选择配置方式。

### {profile}

上图配置中的{profile}，是由别的微服务指定的，如用户中心指定配置，用户中心里会有如下配置，



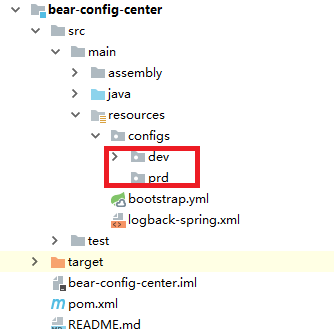
这里的profile: dev就会注入到

配置目录就成了classpath:/configs/dev

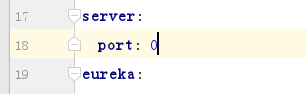
用户中心启动的时候，就会从配置中心拉取配置，目录就是

classpath:/configs/dev/bear-user-center.yml

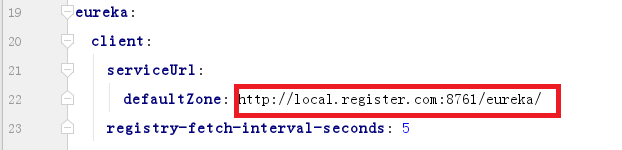
因此用户中心通过配置test或者production等等自定义的字符串，启动时会去找相应的配置，来达到分环境配置的目的，如



### 端口

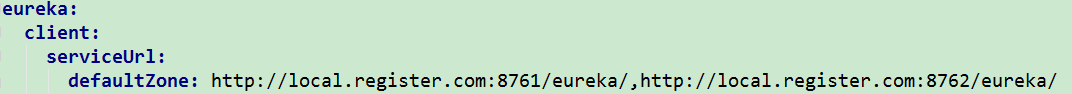
这里配置成0，启动时，项目会随机一个端口号

### 注册到注册中心



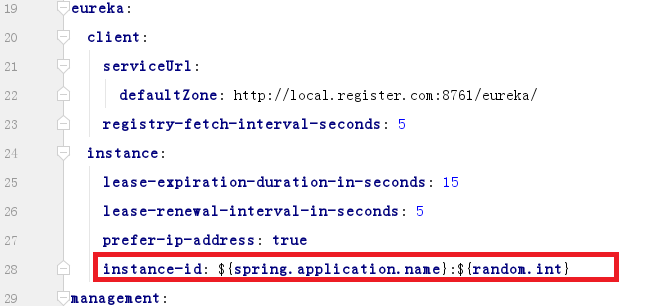
这里需要注意，地址后面有个/eureka/

如果是多注册中心，那么通过逗号分隔



### 注册中心里的显示

因为我们是随机端口号，我们这里用了随机数字来显示

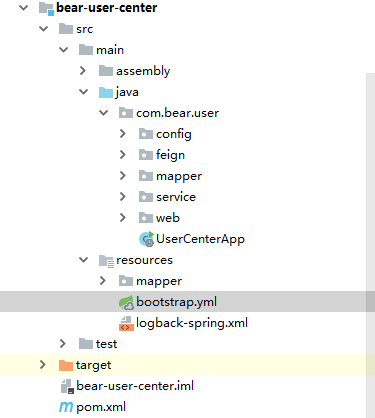


注意那个随机数字，并不是真正的端口号，点击跳转之后

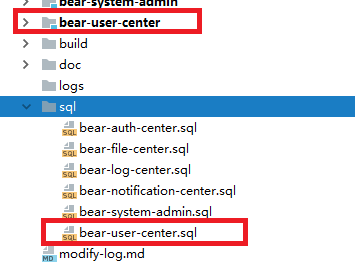
http://192.168.56.8:53484/info

这里我们才能看到真正的端口号

## 用户中心

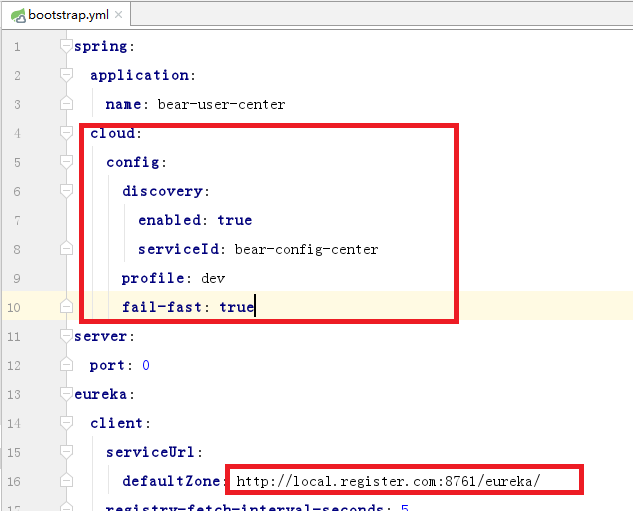


### 数据库脚本



在sql文件夹下bear-user-center.sql里是用户中心的数据脚本，包含建库、建表语句和初始化数据。

### bootstrap.yml

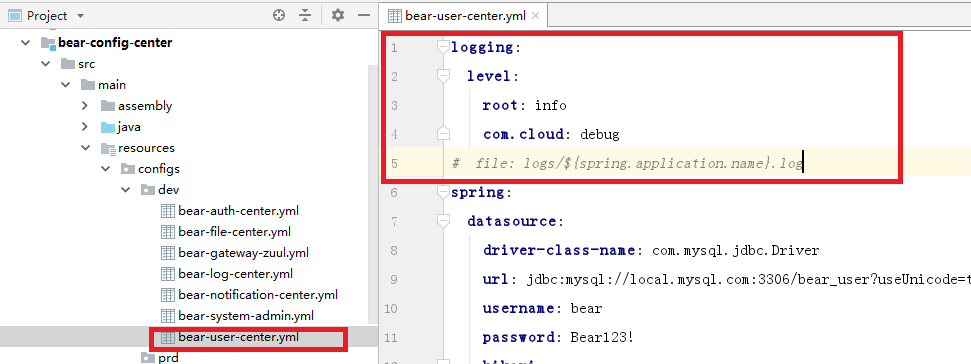


这里主要配置，配置中心的serviceId就是配置中心的spring.application.name，还有自己的profile，还有注册中心的url

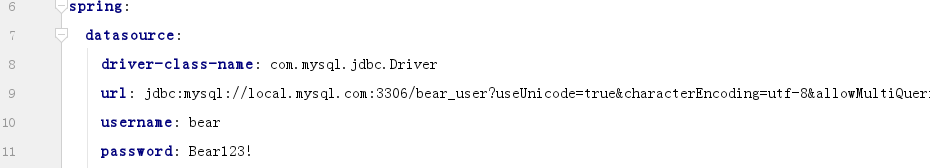
### bear-user-center.yml

这里配置了用户系统具体的一些配置，比如数据库、mq、mybatis、日志级别等。

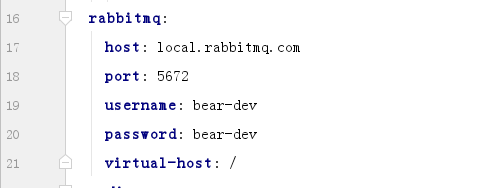
#### 日志级别和文件配置



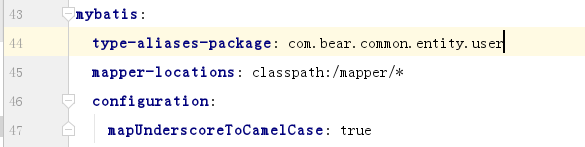
#### 数据源配置



#### Rabbitmq配置



#### Mybatis配置

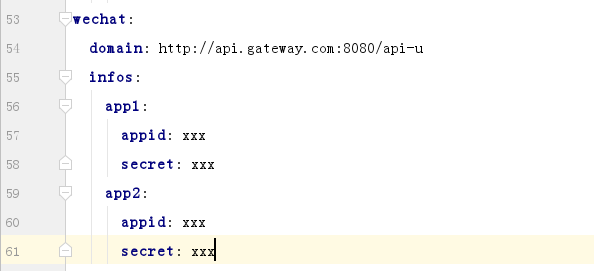


别名包有多个的话，逗号隔开，

复杂sql写在mapper.xml文件里，在下图路径下



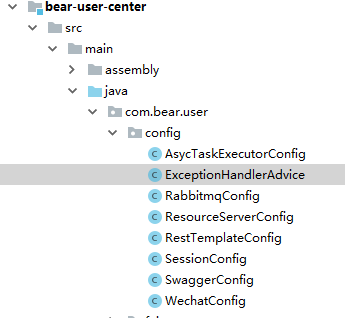
#### 微信公众号配置



详细看下代码com.bear.user.service.impl.WechatServiceImpl和

bears \bear-system-admin\src\main\resources\static\pages\wechat\index.html

### 配置类

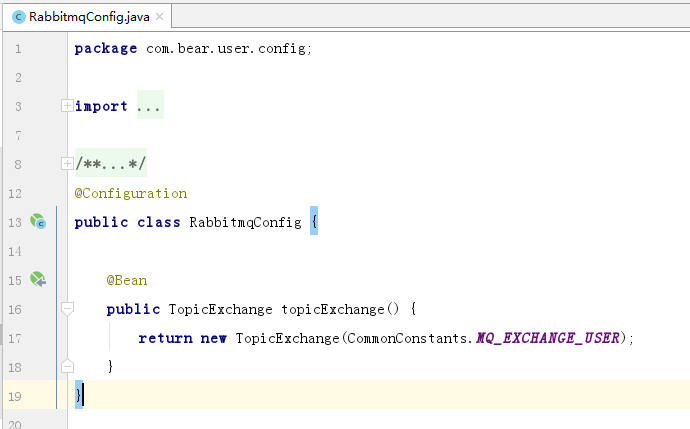


#### 全局异常处理



抛出[java](eclipse-javadoc:%E2%98%82=user-center/D:%5C/Program%20Files%5C/Java%5C/jdk1.8.0_112%5C/jre%5C/lib%5C/rt.jar%3Cjava).[lang](eclipse-javadoc:%E2%98%82=user-center/D:%5C/Program%20Files%5C/Java%5C/jdk1.8.0_112%5C/jre%5C/lib%5C/rt.jar%3Cjava.lang).IllegalArgumentException异常的接口将返回http状态码400

#### Rabbitmq的exchange声明



这里声明一个topic类型的exchange，发消息时用。

#### 资源权限配置



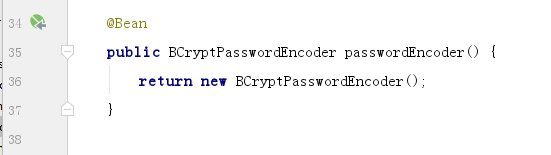
将我们的项目作为资源服务器

是启动权限注解支持



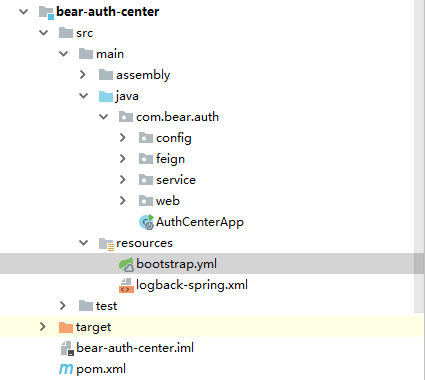
这里符合规则的url将不做权限拦截。

#### 密码加密处理器

上图的

声明一个密码加密和校验处理器Bean，该bean是spring security自带的。

## 认证中心



### 数据库脚本

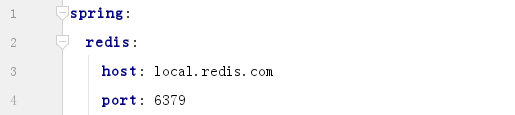
在sql文件夹下bear-auther-center.sql里是认证中心的数据脚本，包含建库、建表语句和初始化数据。

### bootstrap.yml

除spring.application.name是bear-auther-center.外，其他如用户中心的bootstrap.yml相同

### bear-auther-center.yml

#### redis配置

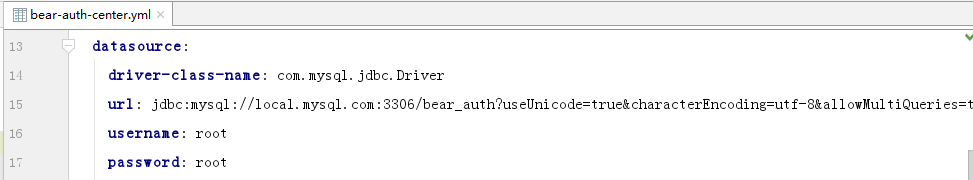


如redis有密码，与host同层级加节点password，如

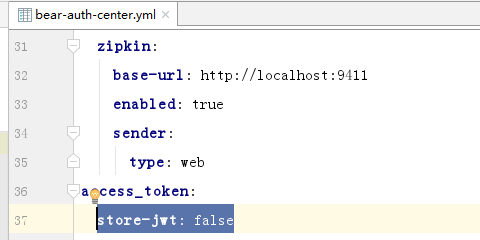


注意password冒号后加一个空格

#### 数据库配置

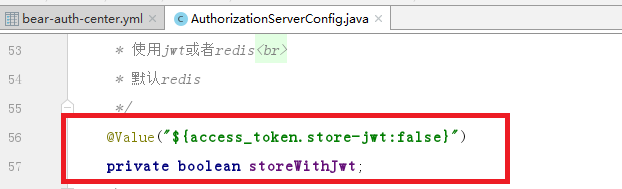


#### token是否用jwt

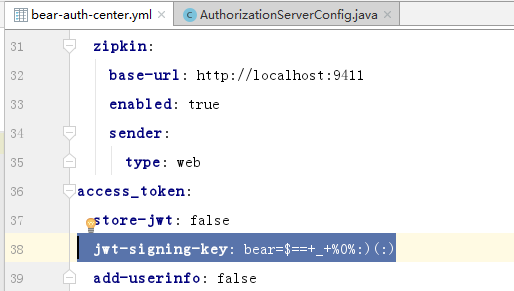


false的话token是默认的uuid，true的话token将采用jwt

**com.bear.auth.config.AuthorizationServerConfig**里面



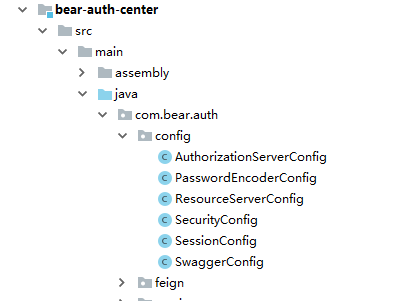


使用jwt时，需要配置这个签名key，具体可看下

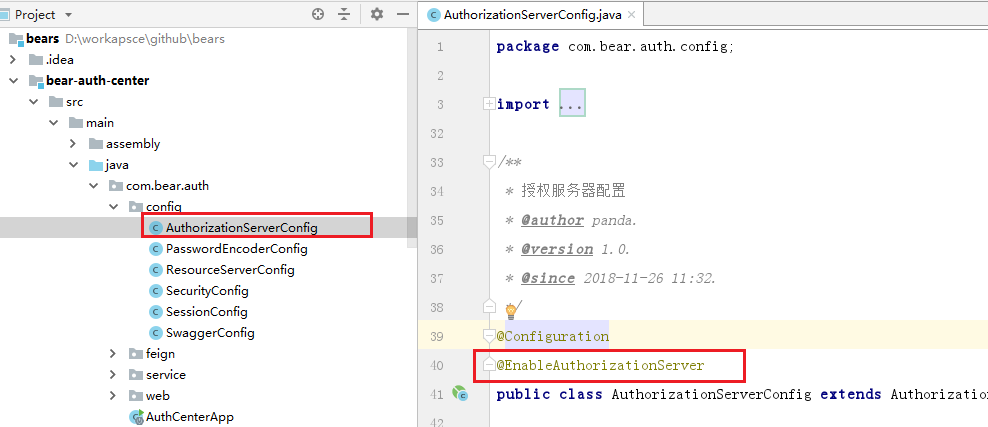
**com.bear.auth.config.AuthorizationServerConfig**里面的



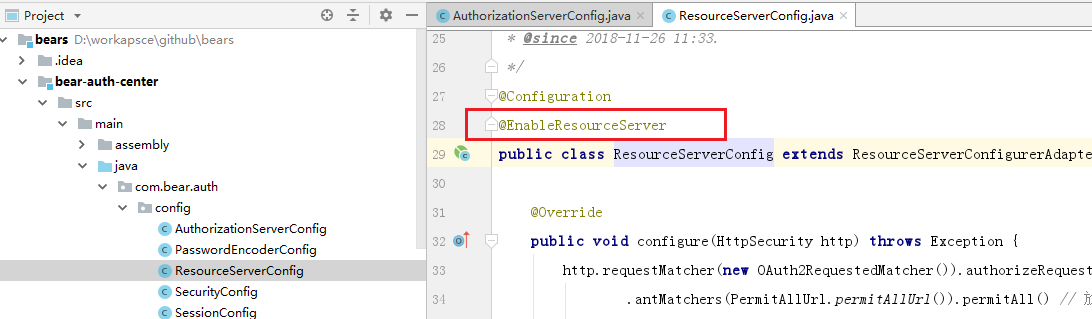
### 配置类



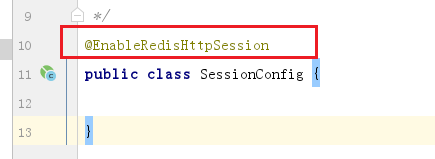
#### 授权服务器配置



#### 资源服务器



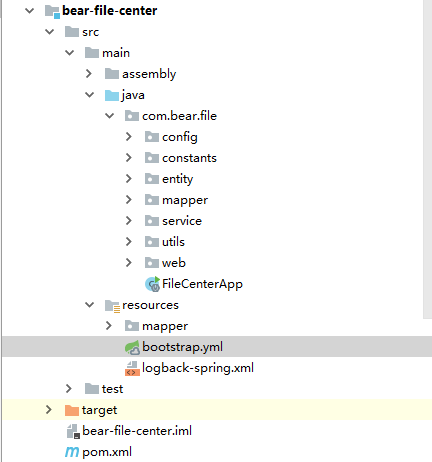
#### Session共享配置



用redis做session共享，在授权码模式下，可能会涉及参数code和state和redirect\_url的传递，多台服务器下需要共享session。

目前该项目没用授权码模式，此处不设置也没问题。

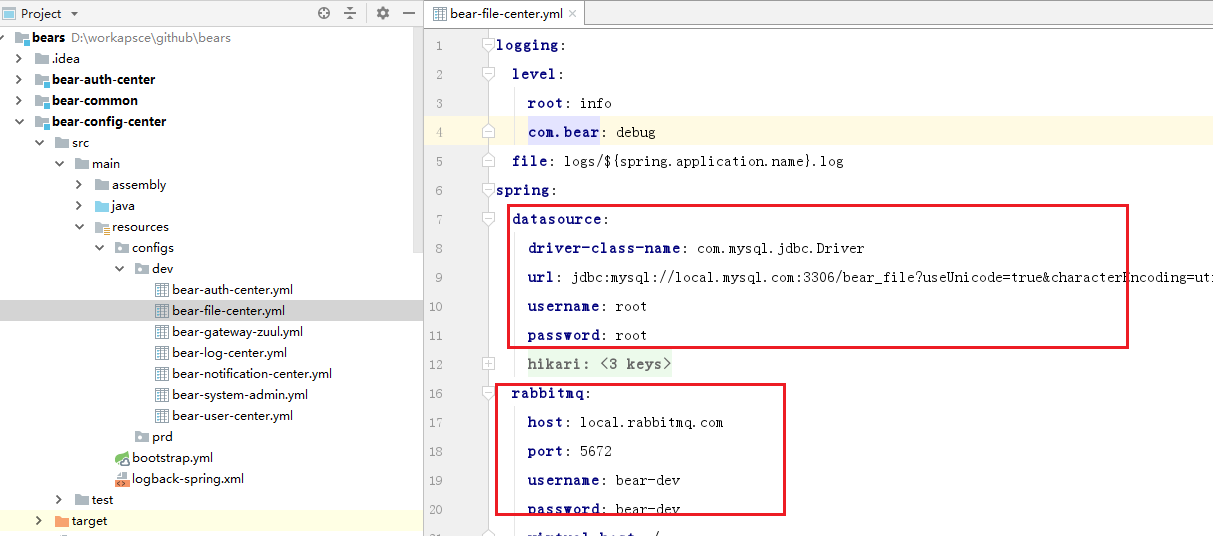
## 文件中心



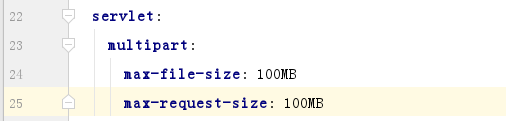
bootstrap.yml里spring.application.name为bear-file-center其余跟用户中心的一样

### bear-file-center.yml

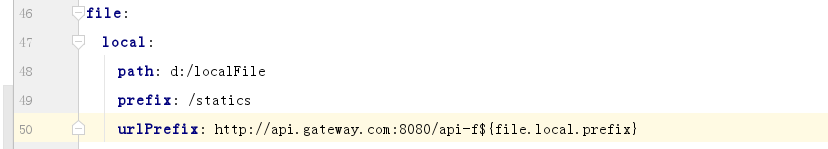
#### 数据库和mq



#### 上传文件大小限制



#### 自定义配置-本地存储文件



上图path是上传文件存储根路径

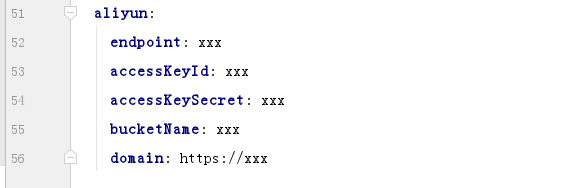
上图prefix是前缀

上图urlPrefix是域名加前缀

如d:/localFile/aaa.png用url访问就是

<http://api.gateway.com:8080/api-f/statics/aaa.png>

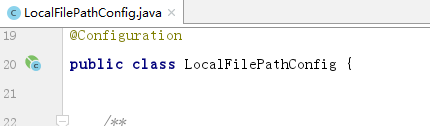
#### 阿里云存储文件



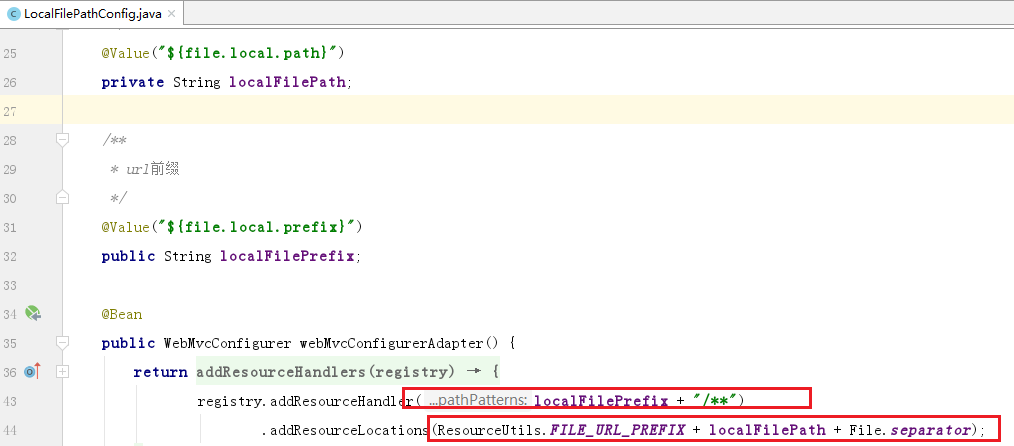
如要上传图片到阿里云，这里需要配置阿里云对象存储OSS相关配置，详细待续….。

### 配置类

#### 加载jar包外文件

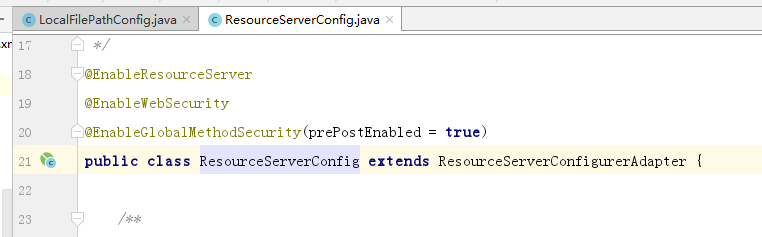


上传文件存储路径肯定是在jar包外部的，这里不像传统war包是解压成文件夹的，因此这里要做个静态资源的映射处理。



这里将url前缀和存储路径做了个映射

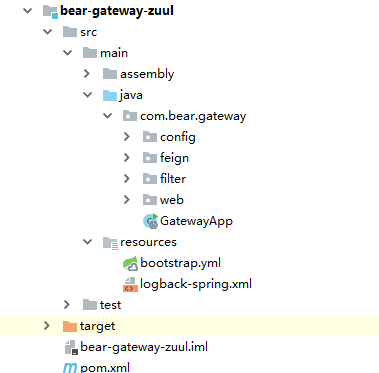
#### 资源服务器



这里要将静态资源下的路径放开权限



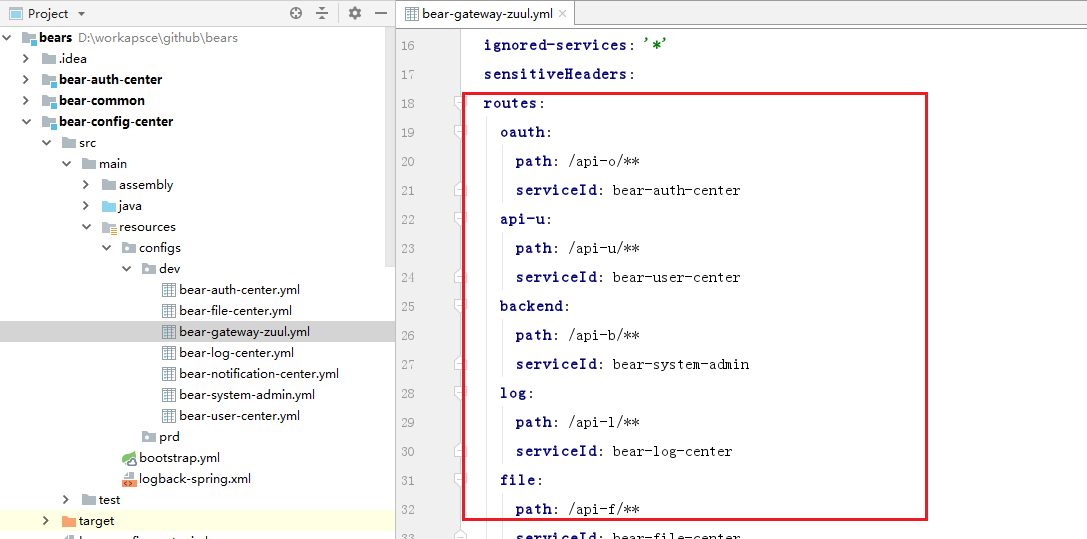
## 网关



bootstrap.yml里spring.application.name为bear-gateway-zuul其余跟用户中心的一样

### Bear-gateway-zuul.yml

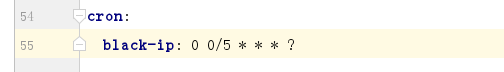
#### 路由规则



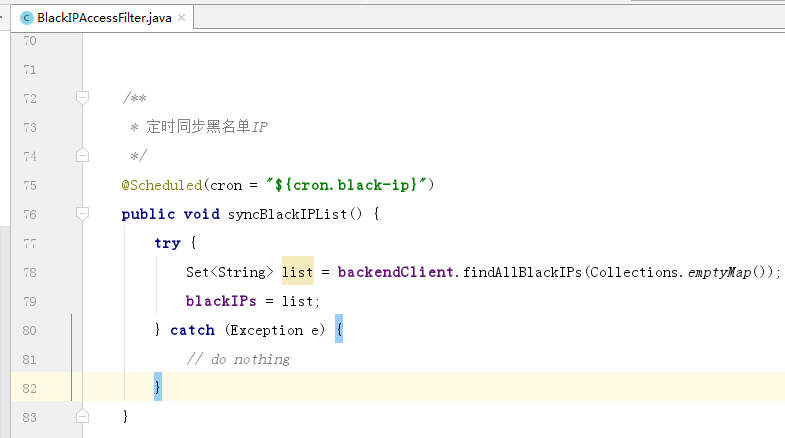
**sensitiveHeaders**过滤客户端附带的headers，如：

sensitiveHeaders: X-ABC  
如果在发请求时带了X-ABC，那么X-ABC不会往下游服务传递。

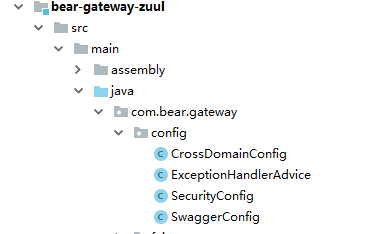
#### 自定义参数



这里有个cron定时任务表达式，每5分钟执行一次，

**com.bear.gateway.filter.BlackIPAccessFilter **

### 配置类

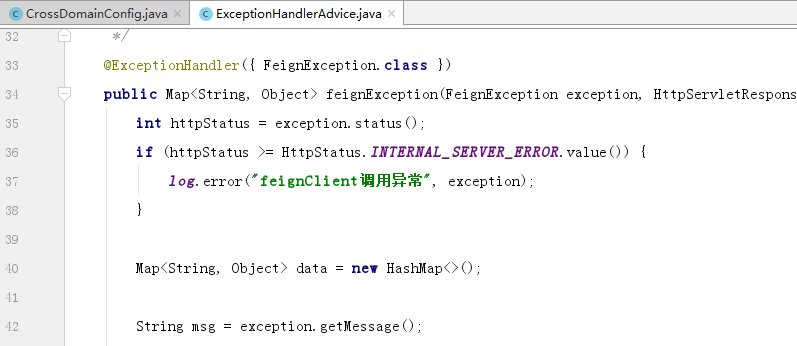


#### 跨域配置



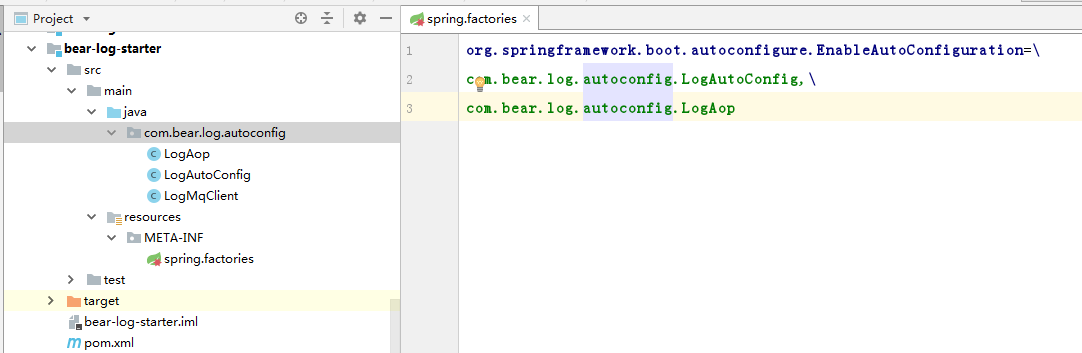
只需要在网关层配置，别的微服务不需要配置跨域

#### 异常处理

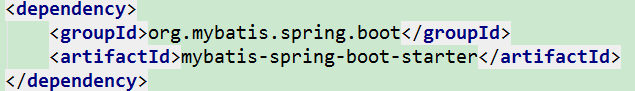


这里主要处理FeignException，这个是feignclient调用时的异常，不处理的话将会抛出500服务端异常，这里只是将下游服务的原始http状态码还原。

## 日志组件log-starter

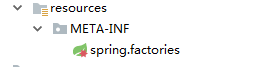


这里是模仿spring boot自动配置写的一个组件，就像spring boot里的各种starter，如

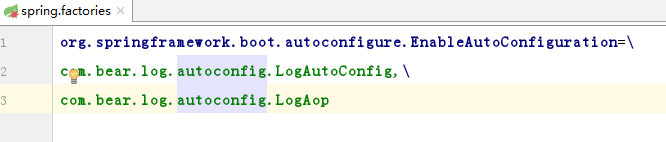


你只需要引入mybatis的starter，和数据源的配置，就可以用mybatis了。

### spring.factories



这里配置自动配置的类

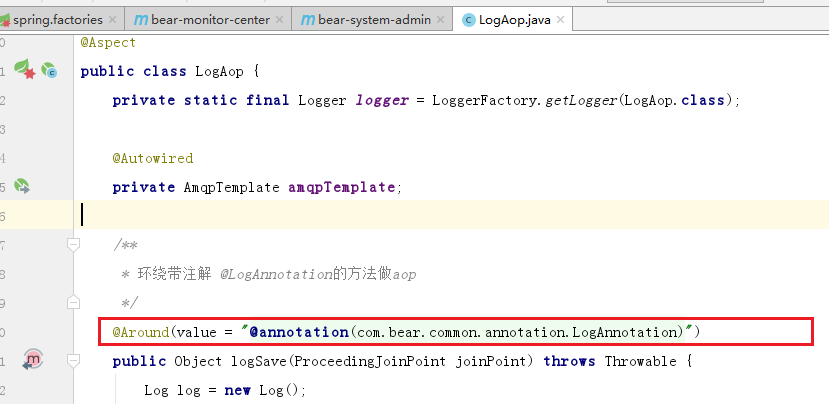


### 使用该组件

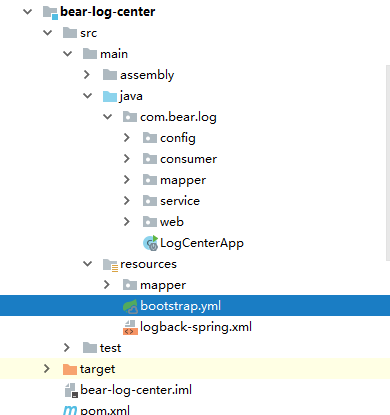
我们这里的bear-log-starter是依赖rabbitmq的，只需要引入



再配置上mq信息，下图的aop类即可生效，就实现了aop日志拦截，将log信息发送到mq队列。

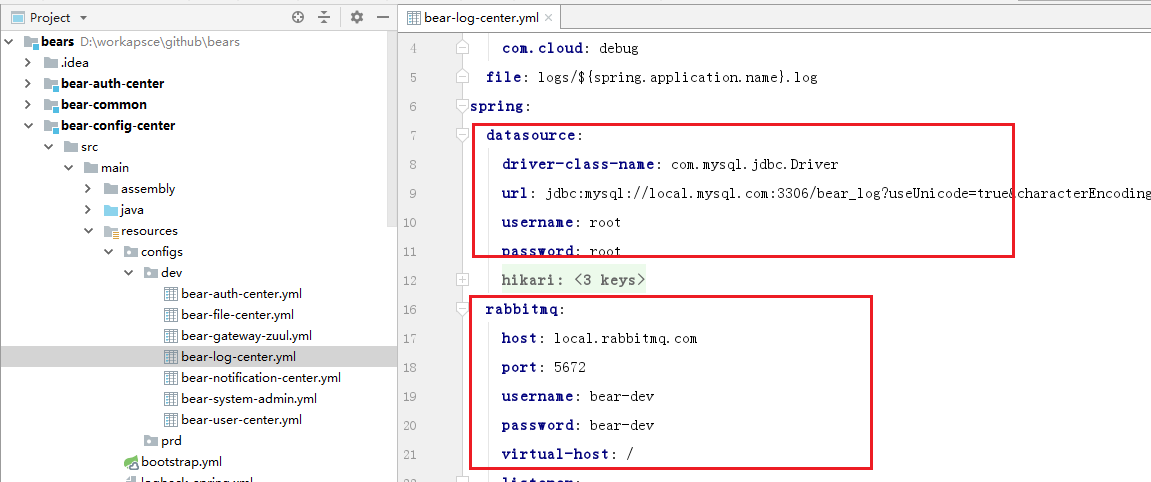


## 日志中心

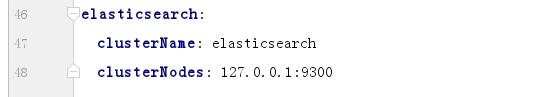


bootstrap.yml里spring.application.name为bear-log-center其余跟用户中心的一样

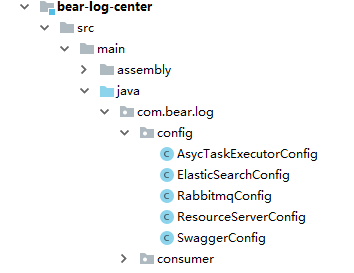
### bear-log-center.yml



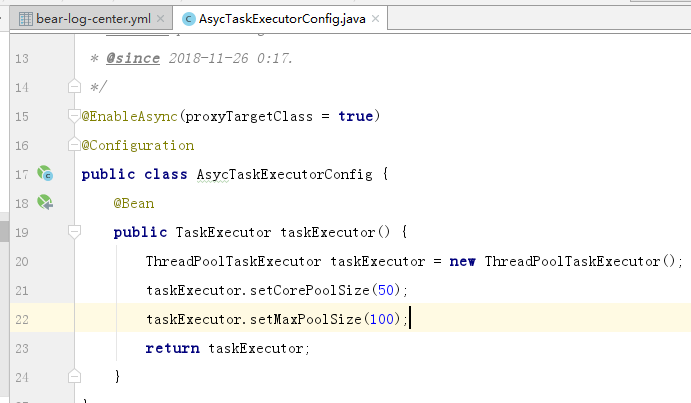
主要是数据库、mq、mybatis的配置，elasticsearch不是必用的



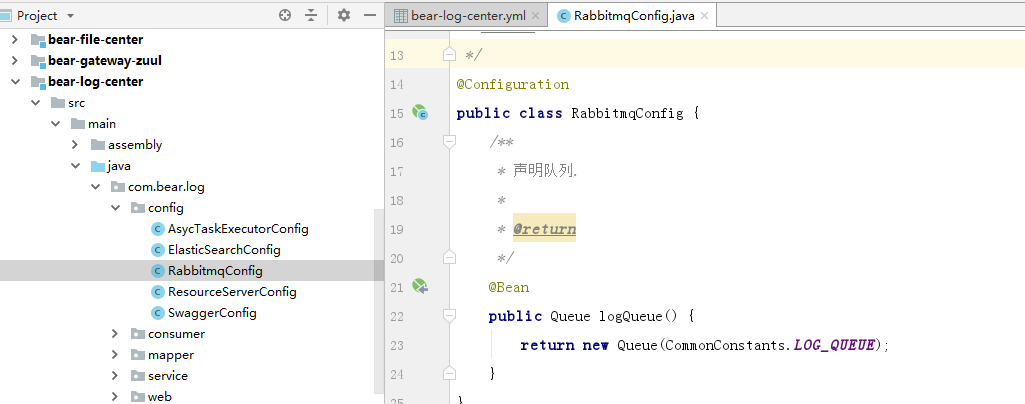
### 配置类



#### 开启异步线程池



#### 声明队列

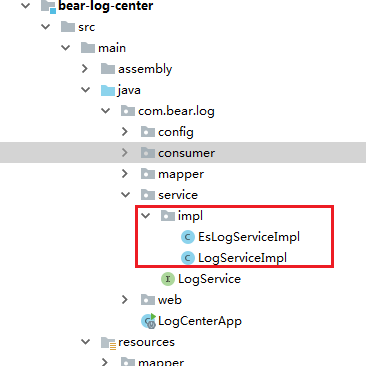


### 处理日志消息

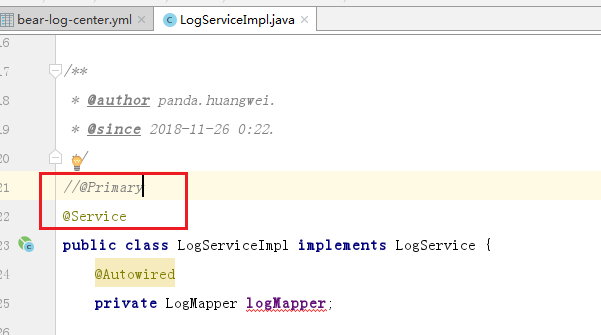


从队列中处理消息，将日志存入数据库

### 日志存储mysql和elasticsearch切换

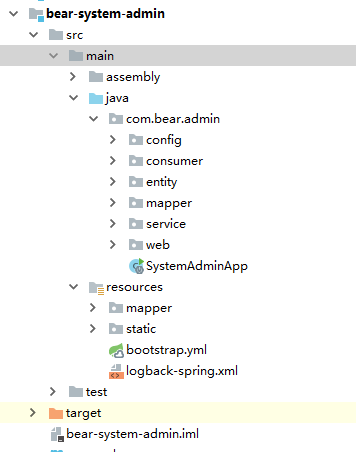


如想存储到elasticsearch的话，

注释掉LogServiceImpl上的@Primary和@Service

或者将@Primary**移到**EsLogServiceImpl上面

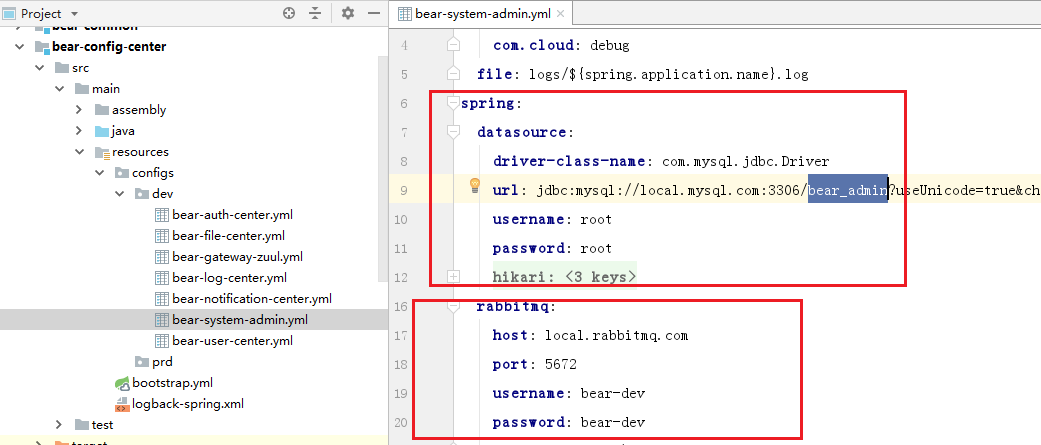
## 后台管理系统



bootstrap.yml里spring.application.name为bear-system-admin其余跟用户中心的一样

### Bear-system-admin.yml

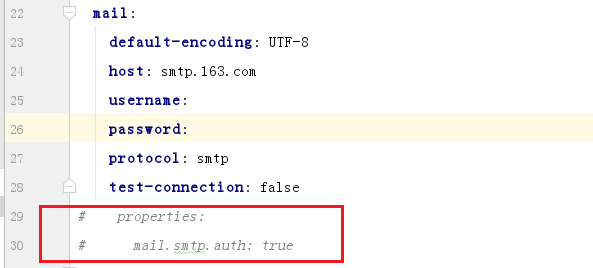
#### 数据库和mq



#### 邮件配置

不发邮件的话，请忽略即可，如要发邮件，

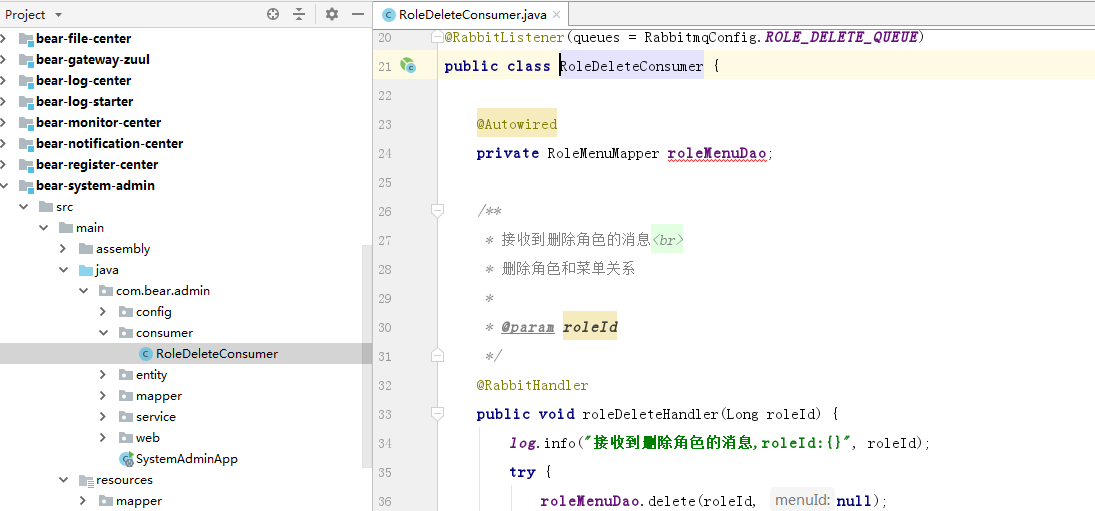
如要使用邮件模块，请写上正确的username和password，并且将最后两行的注释配置打开，否则发邮件可能会失败，如下图



163邮箱如何开启POP3/SMTP/IMAP服务？

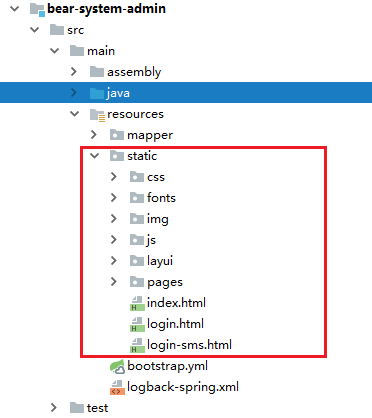
<http://help.163.com/10/0312/13/61J0LI3200752CLQ.html>

### 消息处理

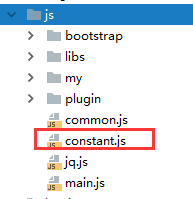


用户系统删除角色时，会抛消息，后台系统将接收该消息，删除菜单与角色的关系

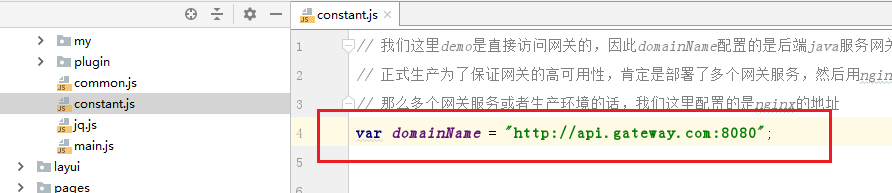
### 静态资源



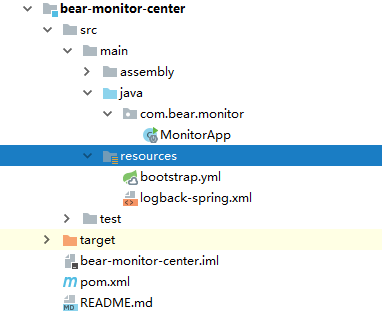
该目录的静态文件和页面是后台管理服务一部分，可单独拿出来部署，



这里定义了一个常量



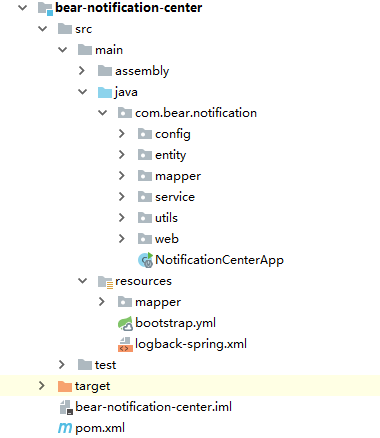
## 监控中心



监控中心我们在bootstrap.yml指定了固定的端口号9001

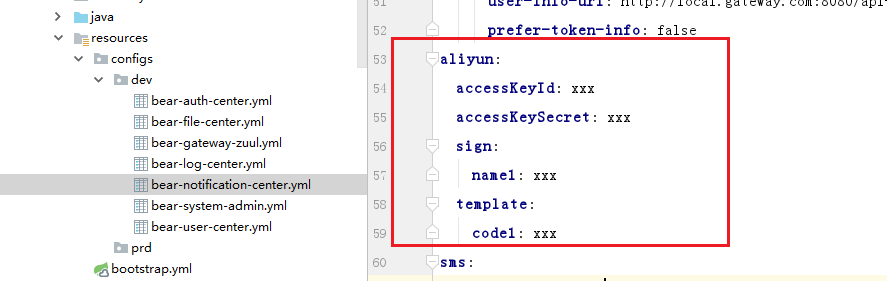
访问http://localhost:9001即可查看监控界面

## 通知中心

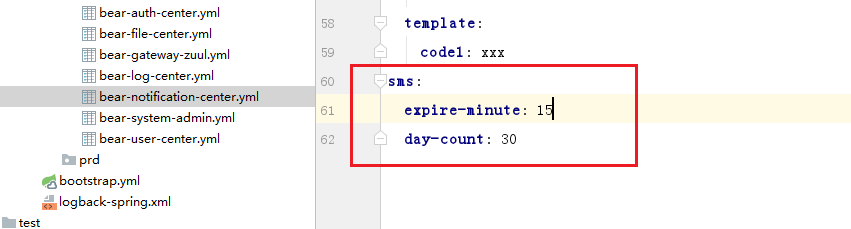


### bear-notification-center.yml

数据库、mq、redis都和别的配置相似，主要说下阿里云短信配置



这些参数需要从阿里云管理控制台自己创建



这里是短信过期时间15分钟，和一天能发送的验证码个数上限。

## Swagger配置

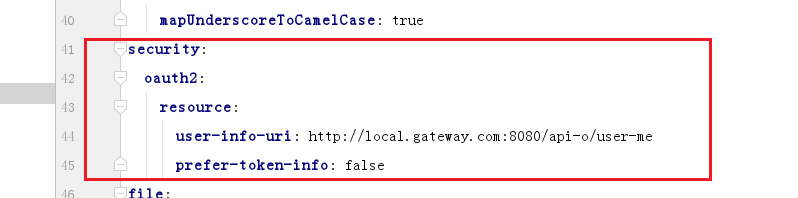
除注册中心、配置中心、监控中心不提供对外接口，别的项目都有

SwaggerConfig这个类，如下图

****

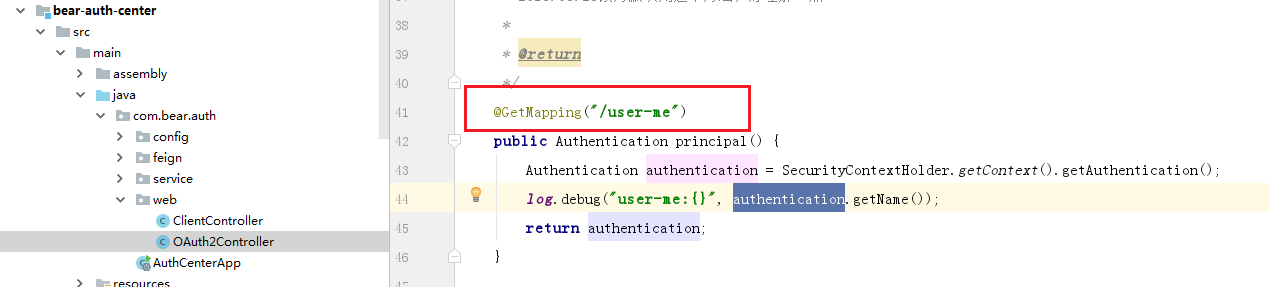
## user-info-uri

在配置中心许多微服务的配置里都有

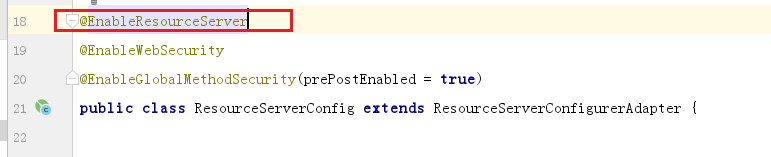


这里是从认证中心获取用户信息，

对应bear-auth-center里的接口，下图



因为加了注解@EnableResourceServer的各微服务都是资源服务器



是需要校验用户权限的，通过user-info-uri可以从认证中心获取到用户的信息和权限，源码

[org](eclipse-javadoc:%E2%98%82=file-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/oauth%5C/spring-security-oauth2%5C/2.0.14.RELEASE%5C/spring-security-oauth2-2.0.14.RELEASE.jar%3Corg).[springframework](eclipse-javadoc:%E2%98%82=file-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/oauth%5C/spring-security-oauth2%5C/2.0.14.RELEASE%5C/spring-security-oauth2-2.0.14.RELEASE.jar%3Corg.springframework).[security](eclipse-javadoc:%E2%98%82=file-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/oauth%5C/spring-security-oauth2%5C/2.0.14.RELEASE%5C/spring-security-oauth2-2.0.14.RELEASE.jar%3Corg.springframework.security).[oauth2](eclipse-javadoc:%E2%98%82=file-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/oauth%5C/spring-security-oauth2%5C/2.0.14.RELEASE%5C/spring-security-oauth2-2.0.14.RELEASE.jar%3Corg.springframework.security.oauth2).[provider](eclipse-javadoc:%E2%98%82=file-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/oauth%5C/spring-security-oauth2%5C/2.0.14.RELEASE%5C/spring-security-oauth2-2.0.14.RELEASE.jar%3Corg.springframework.security.oauth2.provider).[authentication](eclipse-javadoc:%E2%98%82=file-center/C:%5C/Users%5C/wei.zhang1%5C/.m2%5C/repository%5C/org%5C/springframework%5C/security%5C/oauth%5C/spring-security-oauth2%5C/2.0.14.RELEASE%5C/spring-security-oauth2-2.0.14.RELEASE.jar%3Corg.springframework.security.oauth2.provider.authentication).OAuth2AuthenticationProcessingFilter

## bear-common模块

### PermitAllUrl

**com.bear.common.utils.PermitAllUrl**该类主要定义了一些不需要权限拦截的url，



下图这几个是swagger文档需要放开的url，其余是监控中心需要放开的url



### AppUserUtil

**com.bear.common.utils.AppUserUtil**该类是获取当前登录用户的工具类