MicroService

微服务

新手小白总结关于微微服务的知识 欢迎大家讨论

背景:

采用目前主流的微服务系统架构 SpringBoot+SpringCloud+SpringData进行开发

功能模块包括文章、问答、招聘、活动、吐槽、交友、用户中心、搜索中心及第三方登陆等(一个交友博客网站).

特点:

采用前后端分离的模式 模块化设计

根据

2020/5/8

1. 学习微服务开始
2. 关于Docker 的一些基本操作
3. Docker启动服务 systemctl start docker
4. docker ps -a 查看所有的容器
5. docker run -di --name=tensquare\_mysql -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=root centos/mysql-57-centos7
6. docker start xxx 启动某服务

PS: 未能启动mysql微服务显示jdbc错误原因是未在applications.ym jpa 配置上加上

jpa:

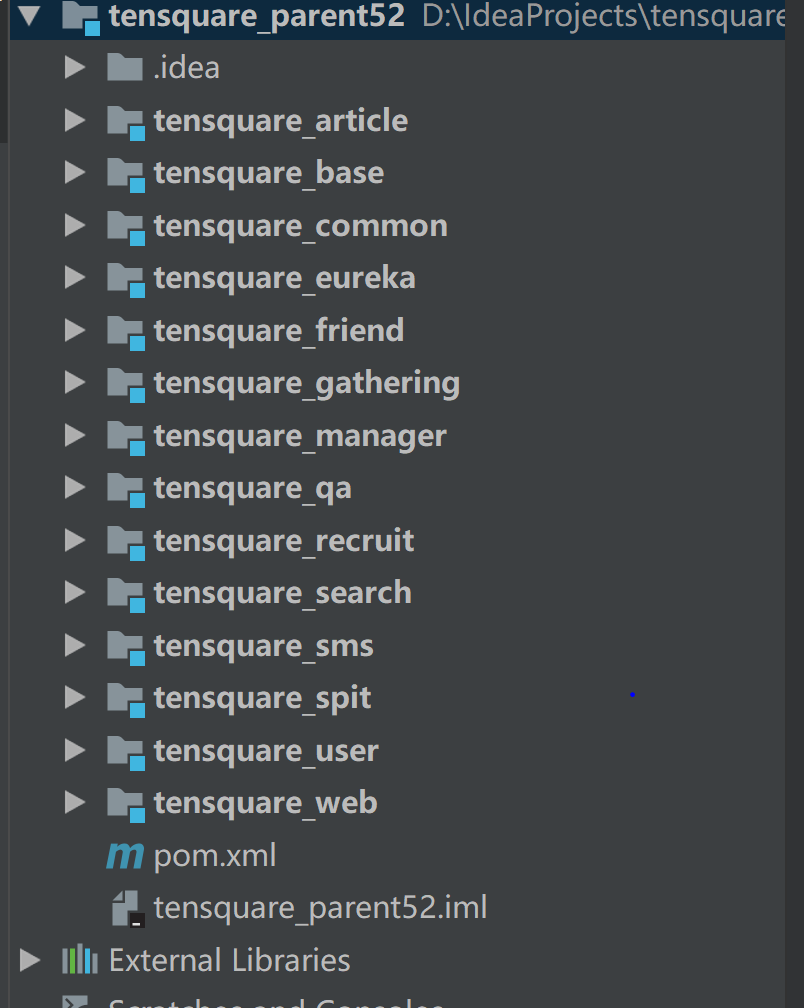
database: mysql

show-sql: true

generate‐ddl: true

(jpa.generate-ddl和jpa.hibernate.ddl-auto都可以控制是否执行datasource.schema脚本，来初始化数据库结构，只要有一个为可执行状态就会执行，比如jpa.generate-ddl:true或jpa.generate-ddl:update，并没有相互制约上下级的关系。)

1. 共18个模块(有一个公共子模块)



tensquare\_common 是父工程

tensquare\_base 是基础工程

1. 采用分库分表设计 每个业务独立成一个数据库
2. API 文档是采用Swagger编写 并与Nignx进行了整合
3. 何为RestFul
4. REST 是Representational State Transfer的缩写，翻译是”表现层状态转化”。 可以总结为一句话：REST是所有Web应用都应该遵守的架构设计指导原则
5. 特征:

面向资源是REST最明显的特征，对于同一个资源的一组不同的操作。资源是服务器 上一个可命名的抽象概念，资源是以名词为核心来组织的，首先关注的是名词。REST要求，必须通过统一的接口来对资源执行各种操作。对于每个资源只能执行一组有限的操作。

1. GET 安全且幂等 只是查询数据，不会影响到资源的变化，因此我们认为它幂等

POST 不安全且不幂等 因为它会对资源本身产生影响，每次调用都会有新的资源产生，因此不满足幂等性。

PUT 不安全但幂等 因为它直接把实体部分的数据替换到服务器的资源，我们多次调用它，只会产生一次影响，但是有相同结果的 HTTP 方法，所以满足幂等性。

DELETE不安全但幂等 调用一次和多次对资源产生影响是相同的，所以也满足幂等性。

**HTTP POST vs HTTP PUT**

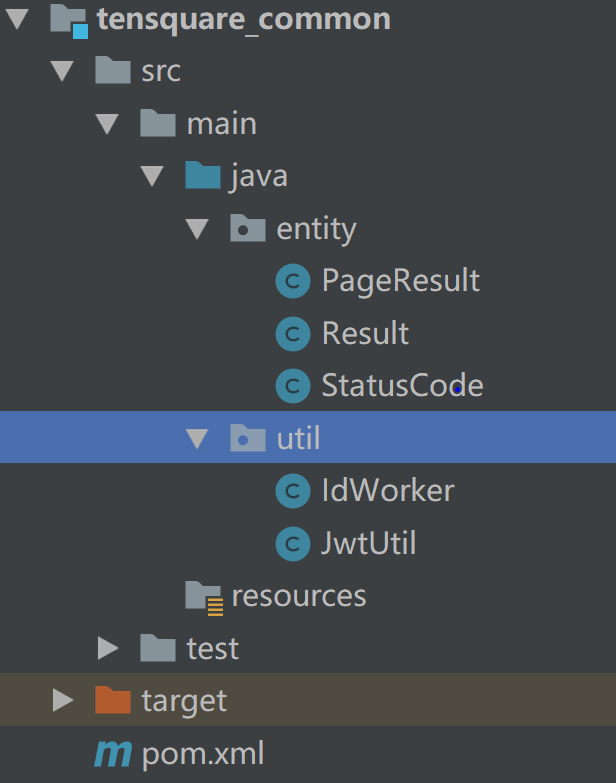
对于 HTTP POST 方法和 HTTP PUT 方法，我们一般的理解是 POST 表示创建资源，PUT 表示更新资源。当然，这个是正确的理解。

但是，实际上，两个方法都用于创建资源，更为本质的差别是在幂等性。HTTP POST 方法是非幂等，所以用来表示创建资源，HTTP PUT 方法是幂等的，因此表示更新资源更加贴切。

PS:HTTP 幂等方法，是指无论调用多少次都不会有不同结果的 HTTP 方法。

不管你调用一次，还是调用一百次，一千次，结果都是相同的。

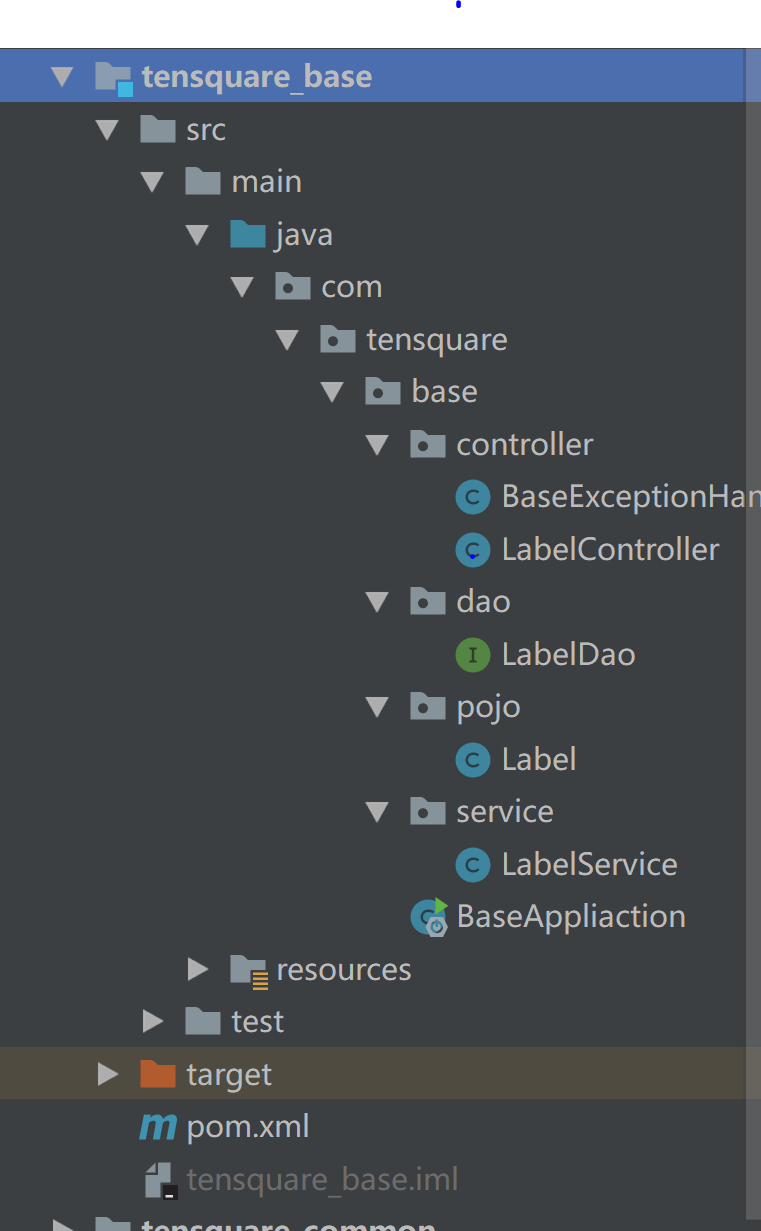
1. 测试工具Postman
2. 搭建第一个模块 公共子模块 tensquare\_common



1. 新建entity包
2. 创建Result类，用于控制器类返回结果
3. 创建类PageResult ，用于返回分页结果
4. 创建类CodeResult ，用于定义返回码

2020/5/9

1. 分布式ID生成器(twitter的snowflake算法)
2. 作用是 数据库主键生成ID
3. 按照时间自增排序
4. Util包下的IdWorker,java
5. 基础微服务-标签CRUD
6. tensquare\_base
7. 模块构成



1. 编写代码过程

* BaseApplication
* 在resources下创建application.yml

server:

port: 9001

spring:

application:

name: tensquare‐base #指定服务名

datasource: driverClassName: com.mysql.jdbc.Driver

url: jdbc:mysql://192.168.184.134:3306/tensquare\_base?characterEncoding=utf‐8

username: root

password: 123456

jpa:

database: MySQL

show‐sql: true

generate‐ddl: true

* 创建com.tensquare.base包，包下创建pojo包 ，包下创建实体类Label

@Entity

@Table(name="tb\_label")

@Id

* com.tensquare.base包下创建dao包，包下创建LabelDao接口

public interface LabelDao extends JpaRepository,JpaSpecificationExecutor{ }

JpaRepository提供了基本的增删改查

JpaSpecificationExecutor用于做复杂的条件查询

* 业务逻辑类

com.tensquare.base包下创建service包，包下创建LabelService类。实现基本的增删改查f功能

直接调用dao层的接口就行

return labelDao.findAll();

return labelDao.findById(id).get();

…

* 控制器类

com.tensquare.user包下创建controller包，创建UserController

@RestController @RequestMapping("/label")

通过 **@PathVariable**可以将 **URL**中占位符参数绑定到控制器处理方法的入参中：URL 中的 {**xxx**} 占位符可以通过@PathVariable(“**xxx**“) 绑定到操作方法的入参中

1. 公共异常处理

在com.tensquare.user.controller包下创建公共异常处理类BaseExceptionHandler

1. 跨域处理

器从一个域名的网页去请求另一个域名的资源时，域名、端口、 协议任一不同，都是跨域

只需要在controller类上添加注解 @CrossOrigin 即可！这个注解其实是CORS的实现。

2020/5/30

1. 查询与缓存(后续会优化)
2. 根据条件查询 修改LabelService ,增加方法

private Specification createSpecification(Map searchMap)

1. 带分页的条件查询 修改LabelService，增加方法

public Page findSearch(Map searchMap,int page,int size)

1. 招聘微服务开发
2. 表结构分析

企业信息和招聘信息

1. 使用代码生成器生成代码

codeutil 来完成代码的生成(生成基本的增删改查操作)

1. 修改Application类名称为RecruitApplication 修改application.yml 中的端口为9002
2. 代码编写

新增加方法的时候从EnterpriseDao新增方法定义开始

public List<Enterprise> findByIshot(String ishot); // where ishot = ? 相当于base->LabelService里面的东西 相比之下简化了很多 在此体现了JPA的强大之处  
//然后到service层里面去 public List<Enterprise> hotList

EnterpriseService新增方法

public List hotlist(){ return enterpriseDao.findByIshot("1"); }

EnterpriseController新增方法

@RequestMapping(value="/search/hotlist",method=RequestMethod.GET) public Result hotlist(){ return new Result(true, StatusCode.OK, "查询成功", enterpriseService.hotlist()); }

1. s
2. S