这里简单说一下怎么用IDEA创建一个Spring Boot微服务项目

# 0.前言

## 0.1使用组件：

Eureka+Zuul+Mybatis+Redis+Lombok

## 0.2简要说明：

Eureka服务注册中心，实现将微服务模块注册到注册中心，这样一个项目的所有微服务都能够被注册和访问，eureka-client注册到eureka-server

Zuul网关，微服务之间虽然能够进行通信，但是每一个微服务都有自己的地址，因此对访问造成困难（地址太多），所以使用网关进行路由转发，这样实现同一个地址能够访问到不同的微服务

Mybatis实现持久层DAO的数据导向

Redis高速缓存

Lombok实体类帮助组件，省却实体类都要写get/set等通用方法的烦恼

## 0.3创建说明：

这里将会创建几个模块在项目下，其中eureka-server和zuul-eureka-client是必要的，前面也说了分别是服务注册中心和路由转发的网关

Eureka-server启动-》监听eureka-client服务来注册-》zuul-client向注册中心注册-》收到第一个注册的client就是zuul网关-》继续接受其他服务前来注册

请求-》zuul网关-》路由转发与负载均衡-》具体的微服务-》响应

## 0.4 示例：

具体服务地址（假定微服务名为Svc\*）：

Svc1: <http://localhost:8081/>

/login

/logout

Svc2:<http://localhost:8082/>

/student

/admin

网关地址：

Zuul: <http://localhost:8888/>

注册中心地址：

Server: <http://localhost:9999/>

0.4.1 启动注册中心

Server: <http://localhost:9999/>

0.4.2 网关注册到注册中心

Zuul: <http://localhost:8888/> 》 Server: <http://localhost:9999/>

0.4.3 其他服务注册到注册中心

Svc1: <http://localhost:8081/> 》 Server: <http://localhost:9999/>

Svc2: <http://localhost:8082/> 》 Server: <http://localhost:9999/>

0.4.4 向Svc1发起/login请求

0.4.4.1 直接访问 Svc1+/login

<http://localhost:8081/login>

0.4.4.2 通过网关访问 Zuul+Svc1+/login

<http://localhost:8888/Svc1/login>

不难看出，在通过网关时，是在网关的基础上加上一层访问路径，这一层访问路径就是具体服务的名称，并且在访问时，注册中心并没有参与进来，仅仅是网关进行了一个路由转发

注册中心存在的意义是：

探测有哪些服务新注册了进来，哪些服务异常了不能使用了，也就是服务判活

[*这里简单说一下怎么用IDEA创建一个Spring Boot微服务项目* 1](#_Toc56615940)

[0.前言 1](#_Toc56615941)

[0.1使用组件： 1](#_Toc56615942)

[Eureka+Zuul+Mybatis+Redis+Lombok 1](#_Toc56615943)

[0.2简要说明： 1](#_Toc56615944)

[0.3创建说明： 1](#_Toc56615945)

[0.4 示例： 1](#_Toc56615946)

[1.创建项目 BootServer 5](#_Toc56615947)

[2.创建模块eureka-server 7](#_Toc56615948)

[2.1 创建一个Spring boot web模块 7](#_Toc56615949)

[2.2首先确认模块已经添加到项目中 10](#_Toc56615950)

[2.3添加eureka server bom依赖 10](#_Toc56615951)

[2.4添加eureka-server使能注解 11](#_Toc56615952)

[2.5配置application.properties文件 12](#_Toc56615953)

[2.6现在你就可以运行eureka-server了 12](#_Toc56615954)

[3.创建模块zuul-eureka-client 14](#_Toc56615955)

[3.1 创建spring boot web模块zuul-eureka-client，创建过程上面有说过 14](#_Toc56615956)

[3.2 添加到BootServer模块，上面有说 14](#_Toc56615957)

[3.3 修改zuul-eureka-client模块的bom.xml文件，添加依赖并同步 14](#_Toc56615958)

[3.4 添加启动入口注解 15](#_Toc56615959)

[3.5 修改application.properties文件 15](#_Toc56615960)

[3.6 选中模块并启动 15](#_Toc56615961)

[4.创建模块test-mybatis 16](#_Toc56615962)

[4.1 创建spring boot web模块 test-mybatis 16](#_Toc56615963)

[4.2 添加bom依赖并同步，记得添加为module 16](#_Toc56615964)

[4.3 添加入口注解，这里因为有测试代码，因此修改扫描包 17](#_Toc56615965)

[4.4 添加application.properties配置 17](#_Toc56615966)

[4.5 现在项目就已经配置好了，可以运行了，也可以在注册中心页面看到此服务 18](#_Toc56615967)

[4.6 简单的使用，我们使用mybatis-dao-service-controller来实现一个全表查询结果， 18](#_Toc56615968)

[4.6.1 先看目录结构： 18](#_Toc56615969)

[4.6.2 给出代码： 19](#_Toc56615970)

[4.7 浏览器访问测试 20](#_Toc56615971)

[4.7.1 直接访问此服务（9001端口） 20](#_Toc56615972)

[4.7.2 通过zuul网关访问 20](#_Toc56615973)

[5.创建模块test-redis 22](#_Toc56615974)

[5.1 创建spring boot web模块test-redis 22](#_Toc56615975)

[5.2 添加bom依赖并更新，注意添加module 22](#_Toc56615976)

[5.3 添加入口注解 22](#_Toc56615977)

[5.4 添加application配置 22](#_Toc56615978)

[5.5 添加测试的Controller 23](#_Toc56615979)

[5.6 启动测试 24](#_Toc56615980)

[5.6.1 直接访问此服务（9001端口） 24](#_Toc56615981)

[5.6.2 通过zuul网关访问 24](#_Toc56615982)

[6.创建模块test-lombok 25](#_Toc56615983)

[6.1 因此首先需要安装插件 25](#_Toc56615984)

[6.2 使用lombok,直接在实体类上加@Data注解即可 25](#_Toc56615985)

[6.3 其他配置和普通的微服务一样配置即可： 26](#_Toc56615986)

[6.3.1 除了需要多一个依赖 26](#_Toc56615987)

[6.3.2 添加application配置 26](#_Toc56615988)

[6.3.3 添加入口注解 27](#_Toc56615989)

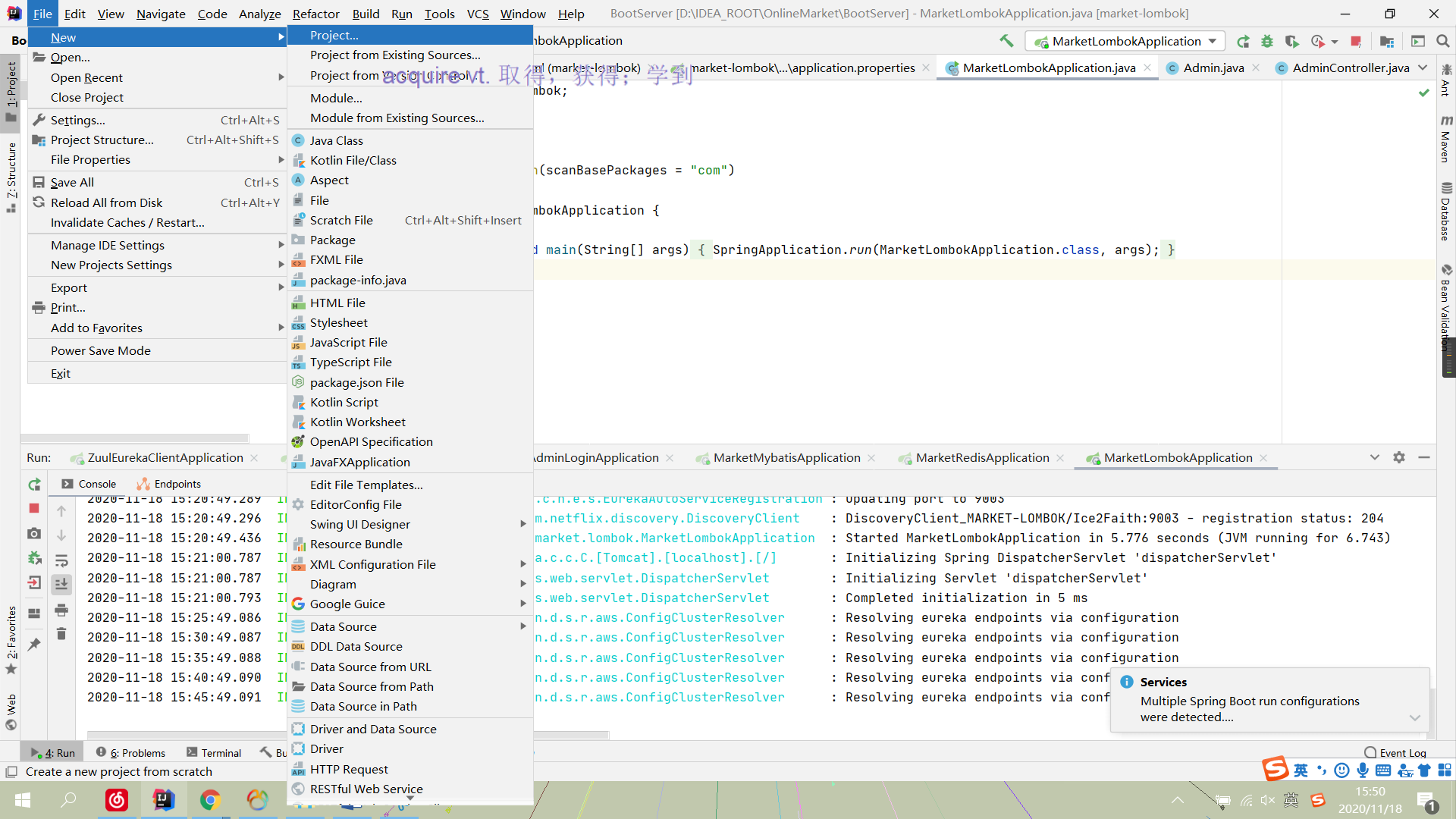
[7.总结 28](#_Toc56615990)

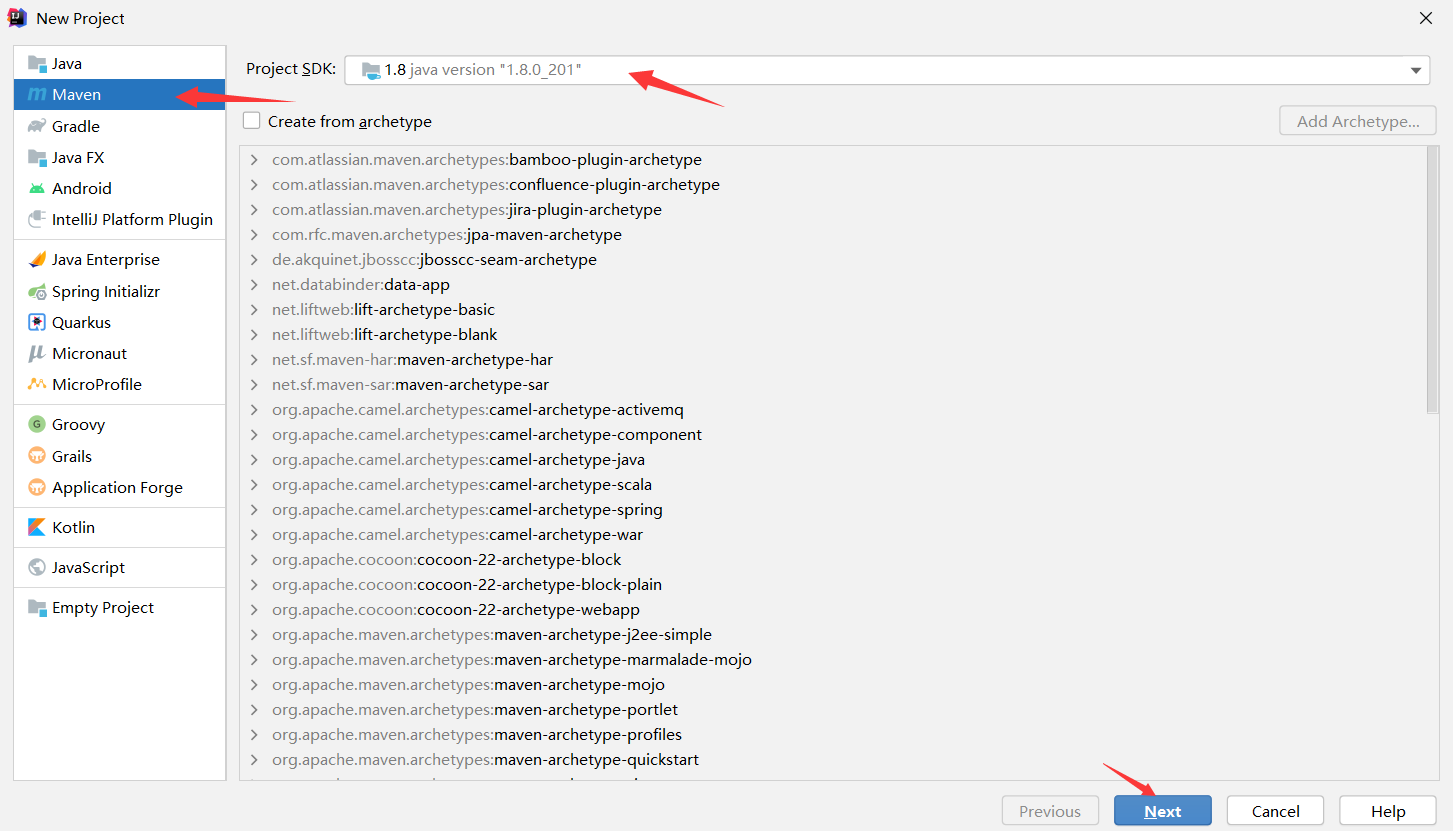
# 1.创建项目 BootServer

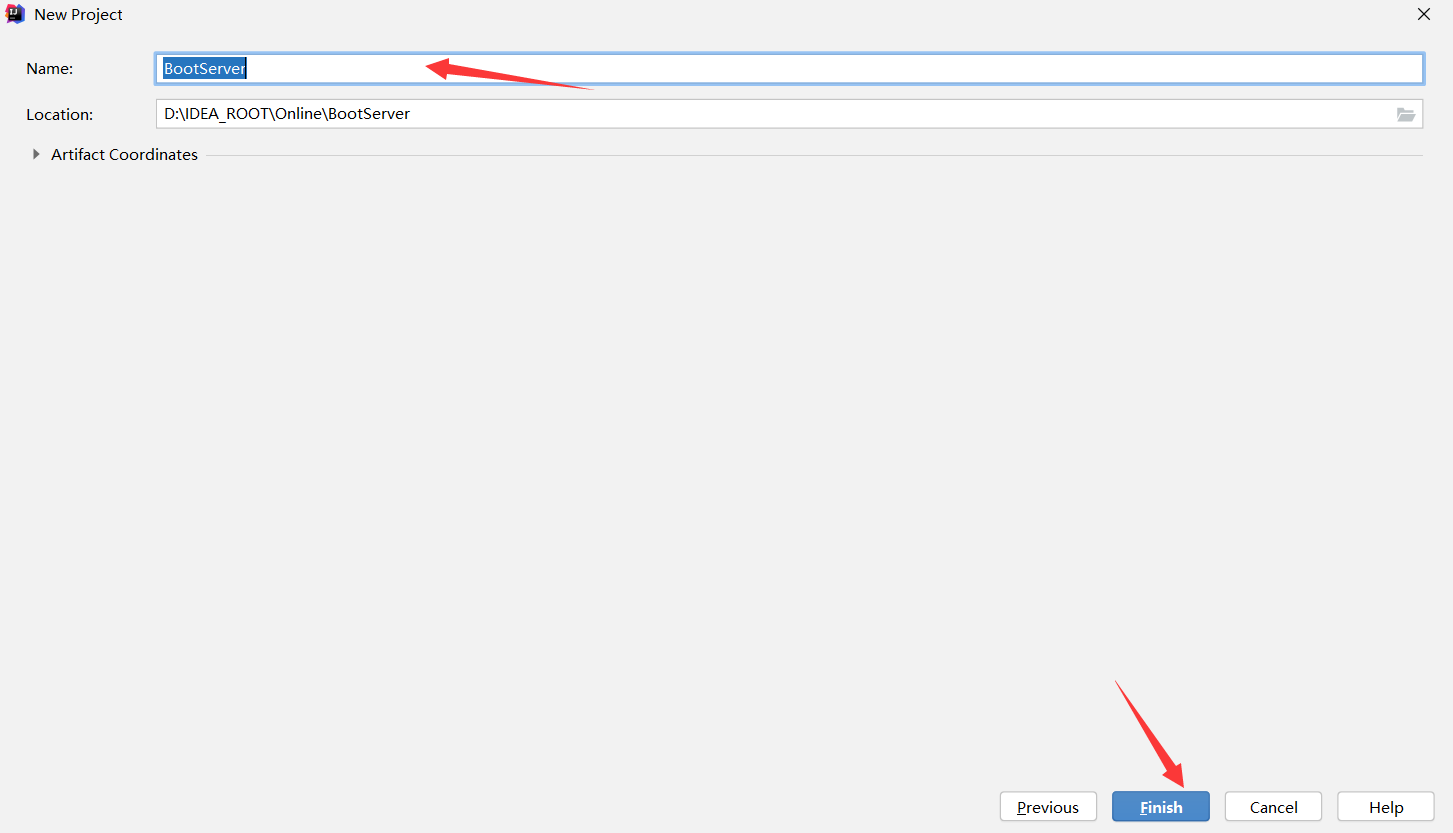
菜单File-new-Project-Maven-Next-填写项目名称和路径-Finish

项目名称我这里就用BootServer，这个名称后面会使用，注意

截图如下：







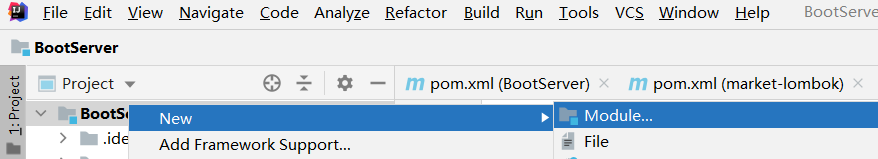
# 2.创建模块eureka-server

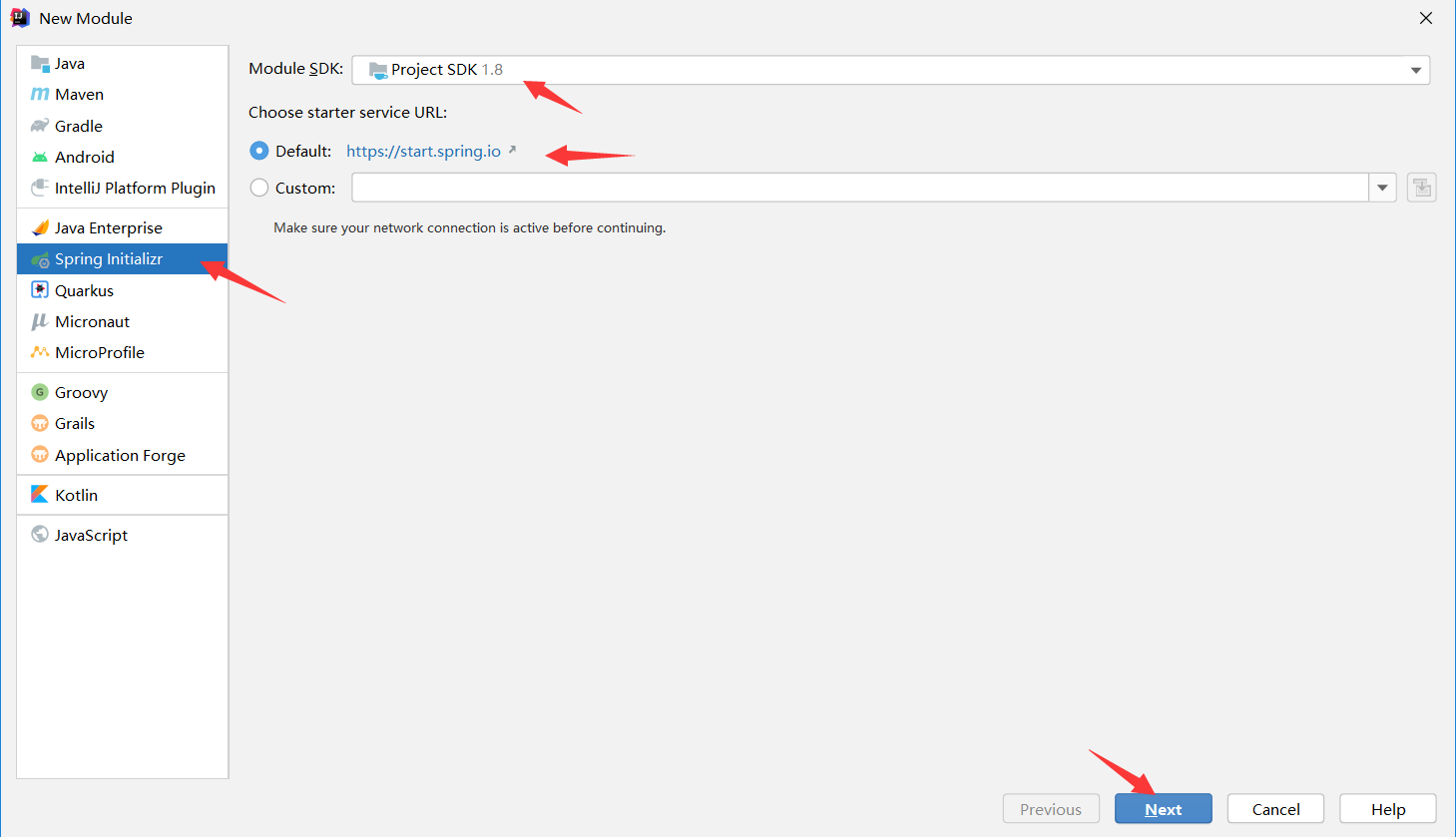
## 2.1 创建一个Spring boot web模块

（这一步只说明一次，后面的模块创建基本都是这个流程）

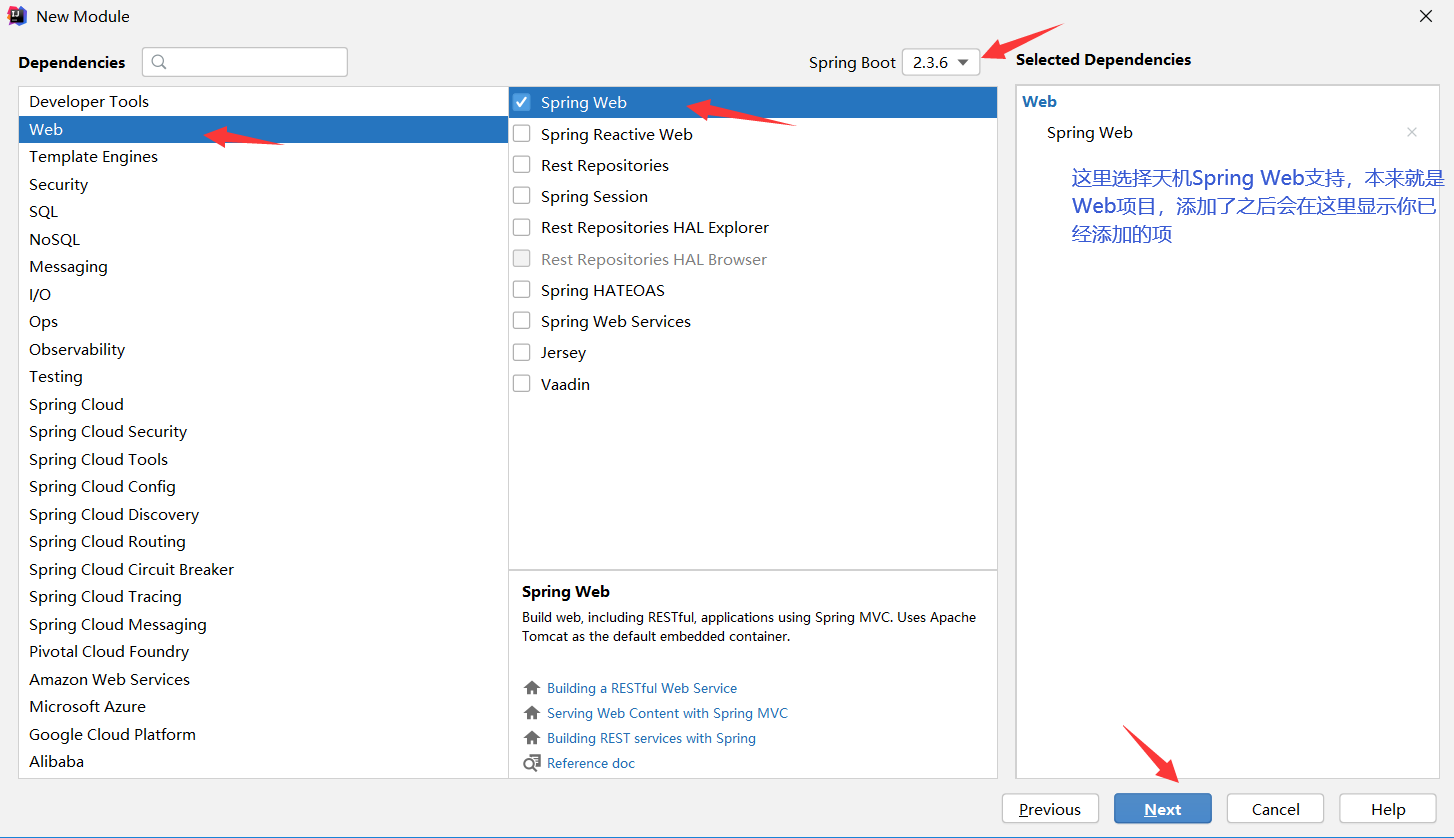
右键刚创建的项目BootServer-new-Module-Spring Initializr-Default-Next

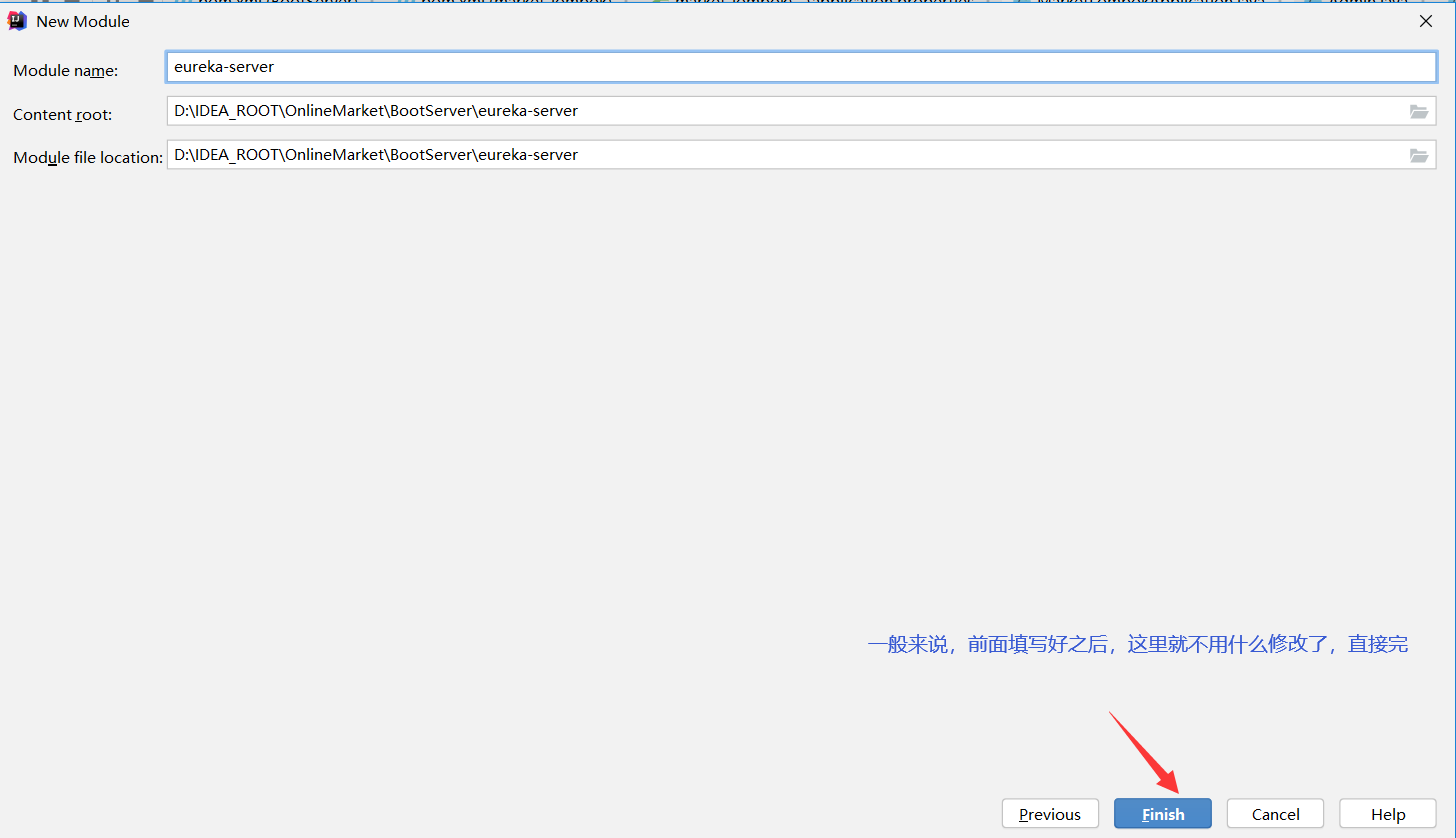
-填写相关信息-Next-添加Web启动支持-Next-确认模块名称-Finish











上面，就完成创建了一个Spring boot web模块，下面添加eureka-server的配置

## 2.2首先确认模块已经添加到项目中

打开BootServer项目下的bom.xml

不出意外的话，是没有模块声明的，我们添加模块，后面添加模块都是这样的，就不再讲



代码：

<modules>  
 <module>eureka-server</module>  
</modules>

## 2.3添加eureka server bom依赖

打开eureka-server模块，找到并打开bom.xml

如下位置添加依赖（dependencies下都可以，不要嵌套进已有的dependency中即可），这个添加依赖步骤类似，后面不再讲

添加之后刷新同步Maven，否则将会无法运行，这可能需要一段时间



代码：

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>  
 <version>2.2.3.RELEASE</version>  
</dependency>

## 2.4添加eureka-server使能注解

在boot入口类上添加注解（这个通用，后面不再讲）

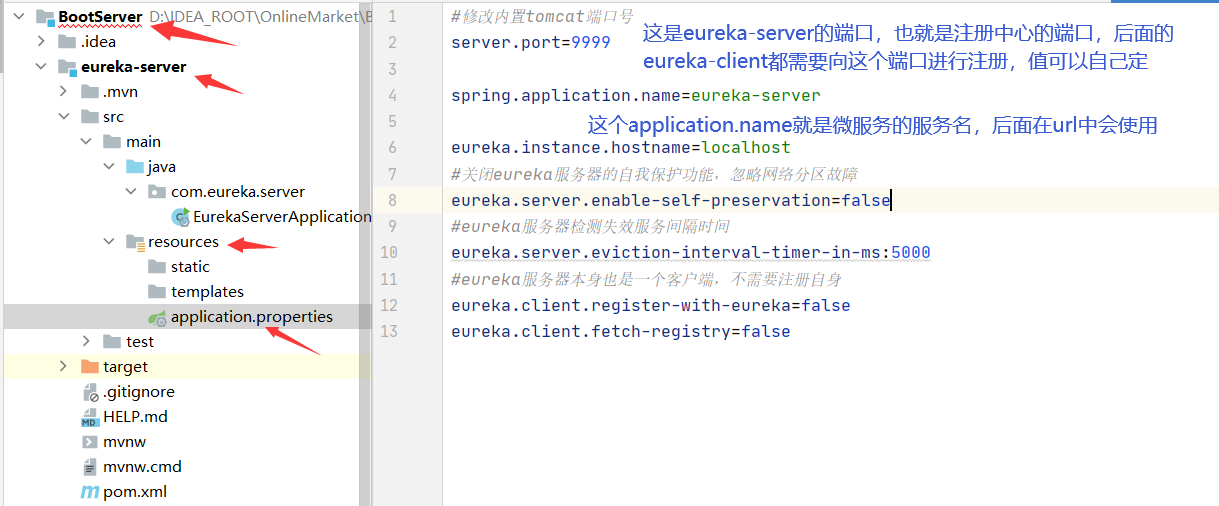


代码：

@EnableEurekaServer

## 2.5配置application.properties文件

这个配置也通用，后面不再讲



代码：

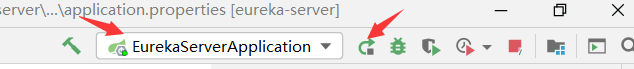
*#修改内置tomcat端口号*server.port=9999  
  
spring.application.name=eureka-server  
  
eureka.instance.hostname=localhost  
*#关闭eureka服务器的自我保护功能，忽略网络分区故障*eureka.server.enable-self-preservation=false  
*#eureka服务器检测失效服务间隔时间*eureka.server.eviction-interval-timer-in-ms:5000  
*#eureka服务器本身也是一个客户端，不需要注册自身*eureka.client.register-with-eureka=false  
eureka.client.fetch-registry=false

## 2.6现在你就可以运行eureka-server了

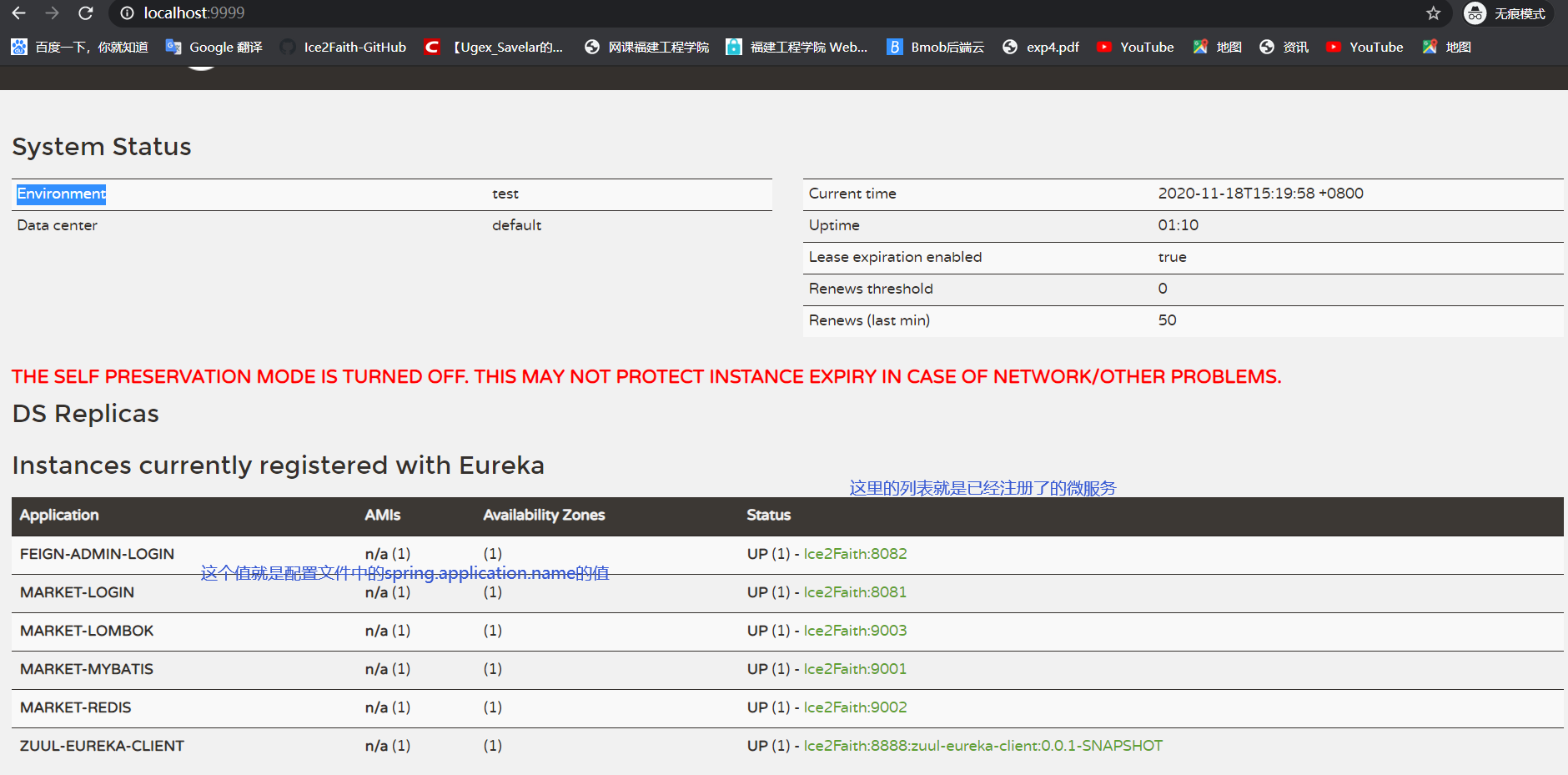
浏览器访问地址：

localhost:9999

这里的9999端口就是我们上面配置的端口，这个地址是服务注册中心的管理页面，这里面可以看到已经注册的服务



浏览器看到的结果：

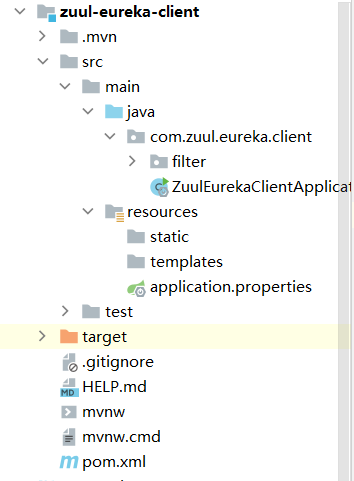


以上就已经创建好了服务注册中心的微服务模块，并成功运行了

你需要看到这个页面有东西，那就成了

# 3.创建模块zuul-eureka-client

最后模块结构：



## 3.1 创建spring boot web模块zuul-eureka-client，创建过程上面有说过

## 3.2 添加到BootServer模块，上面有说

## 3.3 修改zuul-eureka-client模块的bom.xml文件，添加依赖并同步

代码：

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  
 <version>2.2.3.RELEASE</version>  
</dependency>  
  
<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-netflix-zuul</artifactId>  
 <version>2.2.3.RELEASE</version>  
</dependency>

## 3.4 添加启动入口注解

代码：

@EnableZuulProxy  
@EnableEurekaClient

## 3.5 修改application.properties文件

这里说明一下，这个8888端口是我们自己设置的，这个就是以后外网访问的统一网关端口，网关也是微服务，也是需要注册的，所以向注册中心注册，注册中心的URL：<http://localhost:9999/eureka/，其中的9999>端口就是我们的注册中心端口

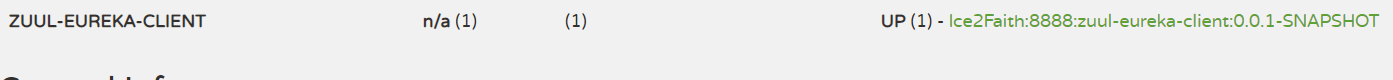
代码：

*#修改内置tomcat端口号*server.port=8888  
  
spring.application.name=zuul-eureka-client  
  
eureka.client.service-url.defaultZone=http://localhost:9999/eureka/  
eureka.client.registry-fetch-interval-seconds=5  
eureka.instance.instance\_id=${spring.cloud.client.hostname}:${server.port}:${spring.application.name}:@project.version@

## 3.6 选中模块并启动

同样去访问注册中心9999端口，应该就能看到一条注册信息了





# 4.创建模块test-mybatis

## 4.1 创建spring boot web模块 test-mybatis

## 4.2 添加bom依赖并同步，记得添加为module

代码：

*<!-- 加载mybatis整合springboot -->*<dependency>  
 <groupId>org.mybatis.spring.boot</groupId>  
 <artifactId>mybatis-spring-boot-starter</artifactId>  
 <version> 2.1.2</version>  
</dependency>  
  
*<!-- MySQL的jdbc驱动包 -->*<dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
</dependency>  
  
*<!-- springboot 开发自动热部署 -->*<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <optional>true</optional>  
</dependency>  
*<!-- alibaba的druid数据库连接池 -->*<dependency>  
 <groupId>com.alibaba</groupId>  
 <artifactId>druid-spring-boot-starter</artifactId>  
 <version>1.1.20</version>  
</dependency>

## 4.3 添加入口注解，这里因为有测试代码，因此修改扫描包



代码：

@SpringBootApplication(scanBasePackages = "com")  
@EnableEurekaClient  
@MapperScan("com.market.dao")  
@EnableTransactionManagement

## 4.4 添加application.properties配置

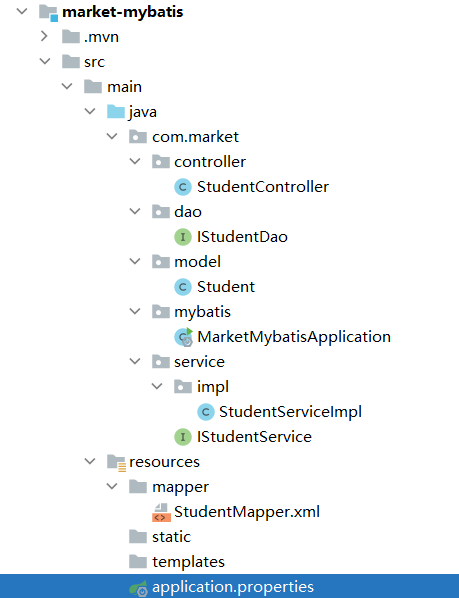
代码：

*#修改内置tomcat端口号*server.port=9001  
  
  
*#负载均衡(ribbon),是zuul中默认集成的，只要这里的application-name值一样，那么将会进行自动的zuul负载均衡  
#也就是访问zuul进行网关转发的时候，会自动进行转发到具体的哪一个同名的微服务中去  
#当然这就要求你的接口要一致，否则还处理个什么*spring.application.name=market-mybatis  
  
eureka.client.service-url.defaultZone=http://localhost:9999/eureka/  
eureka.instance.lease-renewal-interval-in-seconds=5  
eureka.instance.lease-expiration-duration-in-seconds=10  
eureka.instance.instance\_id=${spring.cloud.client.hostname}:${server.port}  
  
*# Mybatis相关配置  
#指定mapper文件的位置*mybatis.mapper-locations=classpath:mapper/\*.xml  
*#指定bean的位置*mybatis.type-aliases-package=com.market.model  
  
*#数据源*spring.datasource.username=root  
spring.datasource.password=ltb12315  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.url=jdbc:mysql://localhost:3306/MyBaitsTestDB?useUnicode=true&characterEncoding=utf8&useSSL=false&serverTimezone=Asia/Shanghai  
spring.datasource.type=com.alibaba.druid.pool.DruidDataSource  
  
*#查看sql语句日志*logging.level.com.market.dao=debug

## 4.5 现在项目就已经配置好了，可以运行了，也可以在注册中心页面看到此服务

4.6 简单的使用，我们使用mybatis-dao-service-controller来实现一个全表查询结果，提供给浏览器显示即可

### 4.6.1 先看目录结构：



### 4.6.2 给出代码：

@RestController  
public class StudentController {  
  
 @Autowired  
 private IStudentService studentService;  
  
 @RequestMapping("getall")  
 public String getall(){  
 List<Student> list=studentService.getAllStudent();  
 String res="all students:";  
 for(Student stu : list){  
 res+="$["+stu.toString()+"]";  
 }  
 return res;  
 }  
}

public interface IStudentDao {  
 List<Student> getAll();  
}

public class Student {  
 private int id;  
 private String name;  
 private int age;  
 private int inYear;  
 private double tall;  
  
  
 @Override  
 public String toString() {  
 return "Student{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", age=" + age +  
 ", inYear=" + inYear +  
 ", tall=" + tall +  
 '}';  
 }  
}

public interface IStudentService {  
 List<Student> getAllStudent();  
}

@Service  
@Transactional  
public class StudentServiceImpl implements IStudentService {  
  
 @Resource  
 private IStudentDao studentDao;  
  
 @Override  
 public List<Student> getAllStudent() {  
 return studentDao.getAll();  
 }  
}

*<?*xml version="1.0" encoding="UTF-8" *?>*<!DOCTYPE mapper  
 PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"  
 "http://mybatis.org/dtd/mybatis-3-mapper.dtd"*>*<mapper namespace="com.market.dao.IStudentDao">  
  
 *<!--查询多条数据-->* <select id="getAll" resultType="student">  
 SELECT id,name,age,inYear,tall FROM Student  
 </select>  
  
  
</mapper>

## 4.7 浏览器访问测试

### 4.7.1 直接访问此服务（9001端口）

<http://localhost:9001/getall>

### 4.7.2 通过zuul网关访问

<http://localhost:8888/test-mybatis/getall>

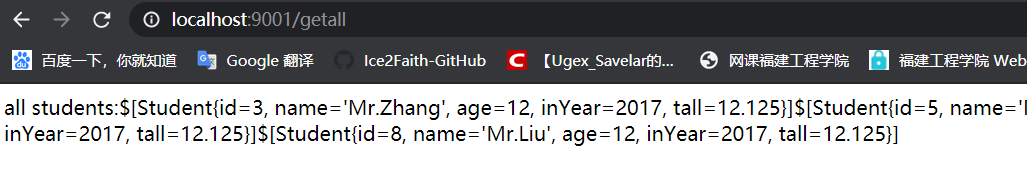
区别：

直接访问：服务的IP端口/服务路径

网关访问：网关的IP端口/服务的application.name/服务路径

服务的application.name在服务的application.properties中配置的

只要你的数据库MyBaitsTestDB中存在表Student和实体类一致，并且有数据，那么你就能在浏览器看到数据，如下



# 5.创建模块test-redis

## 5.1 创建spring boot web模块test-redis

## 5.2 添加bom依赖并更新，注意添加module

代码：

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  
 <version>2.2.3.RELEASE</version>  
</dependency>  
<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-redis</artifactId>  
</dependency>

## 5.3 添加入口注解

@SpringBootApplication(scanBasePackages = "com")  
@EnableEurekaClient

## 5.4 添加application配置

*#修改内置tomcat端口号*server.port=9002  
  
*#负载均衡(ribbon),是zuul中默认集成的，只要这里的application-name值一样，那么将会进行自动的zuul负载均衡  
#也就是访问zuul进行网关转发的时候，会自动进行转发到具体的哪一个同名的微服务中去  
#当然这就要求你的接口要一致，否则还处理个什么*spring.application.name=market-redis  
  
eureka.client.service-url.defaultZone=http://localhost:9999/eureka/  
eureka.instance.lease-renewal-interval-in-seconds=5  
eureka.instance.lease-expiration-duration-in-seconds=10  
eureka.instance.instance\_id=${spring.cloud.client.hostname}:${server.port}  
  
*# Redis数据库索引（默认为0）  
# spring.redis.database=0  
# Redis服务器地址*spring.redis.host=127.0.0.1  
*# Redis服务器连接端口*spring.redis.port=6379  
*# Redis服务器连接密码（默认为空）*spring.redis.password=ltb12315  
*# 连接池最大连接数（使用负值表示没有限制）*spring.redis.pool.max-active=50  
*# 连接池最大阻塞等待时间（使用负值表示没有限制）*spring.redis.pool.max-wait=-1  
*# 连接池中的最大空闲连接*spring.redis.pool.max-idle=10  
*# 连接池中的最小空闲连接*spring.redis.pool.min-idle=0  
*# 连接超时时间（毫秒）*spring.redis.timeout=1000

## 5.5 添加测试的Controller

代码：

@RestController

public class RedisHostController {

@Autowired

private RedisTemplate<String,String> redisTemplate;

@RequestMapping("/redis")

public String redisHost(){

String tk="tkey";

String ss="redis success value";

redisTemplate.opsForValue().set(tk,ss);

ss=""+redisTemplate.hasKey(tk);

ss=redisTemplate.opsForValue().get(tk);

redisTemplate.delete(tk);

ss+=redisTemplate.hasKey(tk);

return ss;

}

}

## 5.6 启动测试

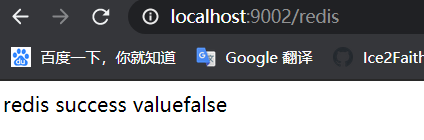
### 5.6.1 直接访问此服务（9001端口）

<http://localhost:9002/redis>

### 5.6.2 通过zuul网关访问

<http://localhost:8888/test-redis/>redis

看到如下显示，即为成功

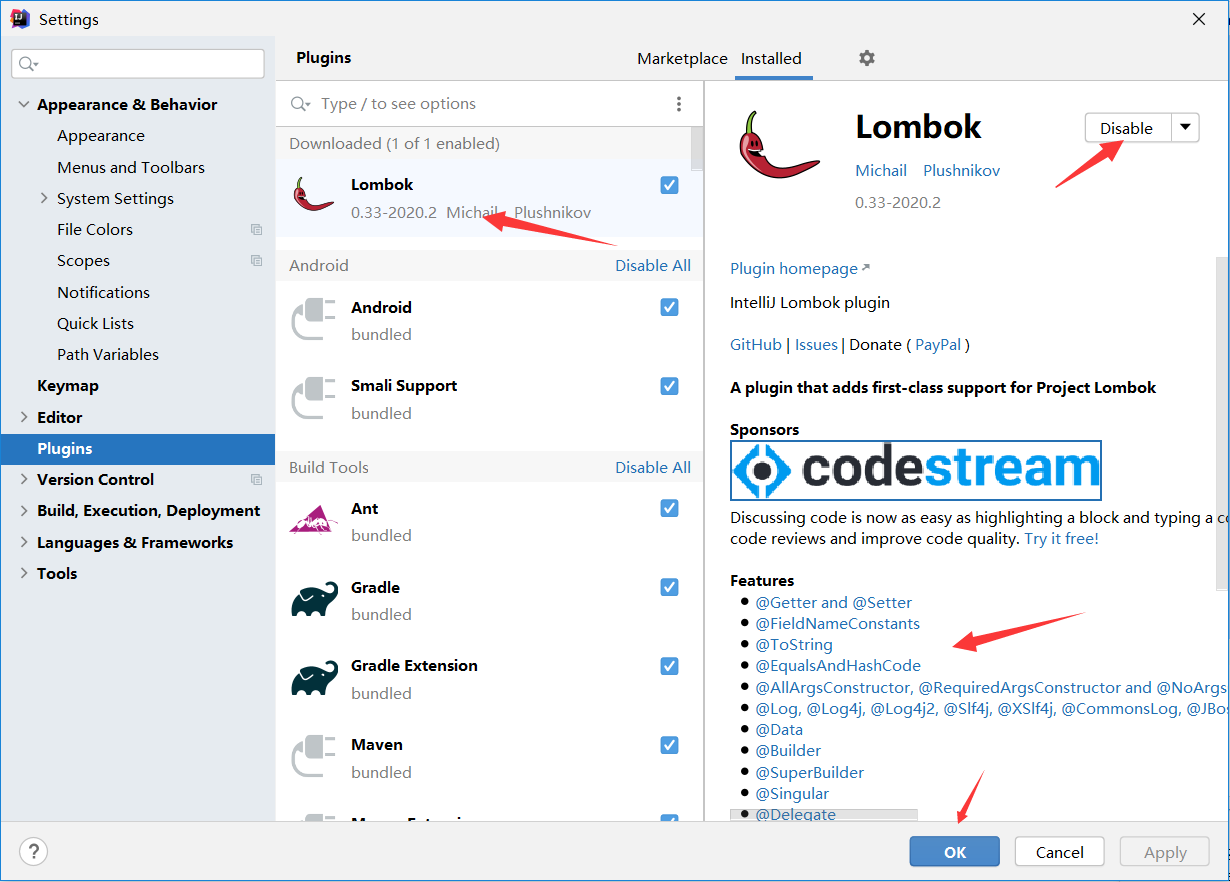


# 6.创建模块test-lombok

Lombok其实是IDEA的一个插件，作用是自动给实体类添加get/set方法，equals方法，tostring方法等

## 6.1 因此首先需要安装插件

菜单File-Setting-Plugins-搜索lombok-选中install-OK



## 6.2 使用lombok,直接在实体类上加@Data注解即可

如下，他就已经有get/set等方法了

@Data  
public class Admin {  
 private int id;  
 private String name;  
 private double salary;  
  
 public Admin(int id, String name, double salary) {  
 this.id = id;  
 this.name = name;  
 this.salary = salary;  
 }  
}

## 6.3 其他配置和普通的微服务一样配置即可：

### 6.3.1 除了需要多一个依赖

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  
 <version>2.2.3.RELEASE</version>  
</dependency>  
<dependency>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
</dependency>

### 6.3.2 添加application配置

*#修改内置tomcat端口号*server.port=9003  
  
  
*#负载均衡(ribbon),是zuul中默认集成的，只要这里的application-name值一样，那么将会进行自动的zuul负载均衡  
#也就是访问zuul进行网关转发的时候，会自动进行转发到具体的哪一个同名的微服务中去  
#当然这就要求你的接口要一致，否则还处理个什么*spring.application.name=market-lombok  
  
eureka.client.service-url.defaultZone=http://localhost:9999/eureka/  
eureka.instance.lease-renewal-interval-in-seconds=5  
eureka.instance.lease-expiration-duration-in-seconds=10  
eureka.instance.instance\_id=${spring.cloud.client.hostname}:${server.port}

### 6.3.3 添加入口注解

@SpringBootApplication(scanBasePackages = "com")  
@EnableEurekaClient

# 7.总结

通过上面的过程，虽然只是单独的某个组件的集成，但是后面就可以进行综合的继承了，比如一个spring boot web中就添加Mybatis,redis,lombok等等的组件