今日任务完成12

@EnableAutoConfiguration 注解

@EnableAutoConfiguration注解内部使用@import(AutoConfigurationImportSelector.class)来加载配置类

配置文件位置：META-INF/spring.factories,该配置文件中定义了大量的配置类，当SpringBoot应用启动时，会自动加载这些配置类，初始化Bean

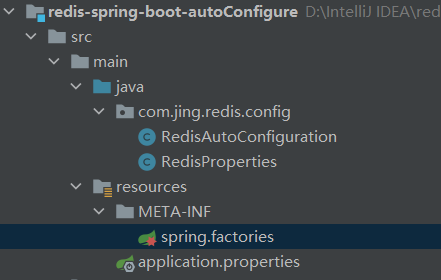
并不是所有的Bean都会被初始化，在配置类中使用Condition来加载满足的条件的Bean

SpringBoot自动配置

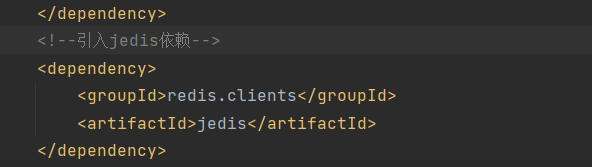
案例需求：自定义redis-starter。要求当导入redis坐标时,SpringBoot自动创建Jedis的Bean

实现步骤：

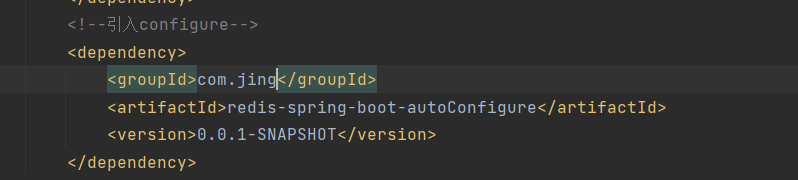
1. 创建redis-spring-boot-autoconfigure模块
2. 创建redis-spring-boot-starter模块，依赖redis-spring-boot-autoconfigure的模块
3. 在redis-spring-boot-autoconfigure模块中初始化Jedis的Bean.定义META-INF/spring.factories文件
4. 在测试模块中引入自定义的redis-starter依赖，测试获取Jedis的Bean，操作redis
5. 创建redis-spring-boot-autoconfigure模块



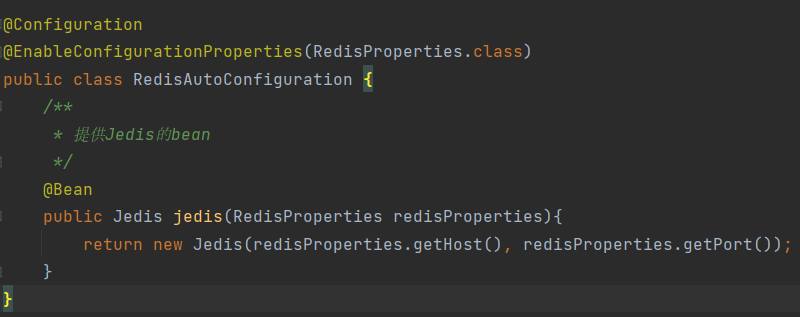


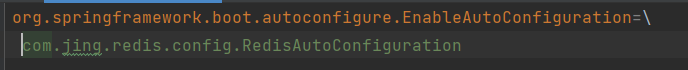


2.创建redis-spring-boot-starter模块，依赖redis-spring-boot-autoconfigure的模块

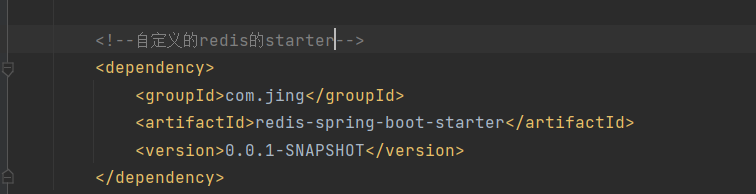


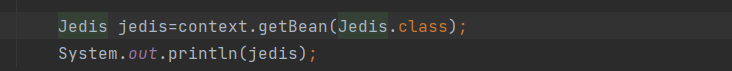
3.在redis-spring-boot-autoconfigure模块中初始化Jedis的Bean.定义META-INF/spring.factories文件





4.在测试模块中引入自定义的redis-starter依赖，测试获取Jedis的Bean，操作redis





SpringBoot监听机制

Java监听机制

SpringBoot的监听机制，其实是对Java提供的事件监听机制的封装

Java中的事件监听机制定义了以下角色：

1. 事件:Event，继承java.util.EventObject类的对象
2. 事件源：Source,任意对象Object
3. 监听器：Listener,实现java.util.EventListener接口的对象

SpringBoot在项目启动时，会对几个监听器进行回调，我们可以实现这些监听器接口，在项目启动时完成一些操作。

ApplicationContextInitalizer、SpringApplicationRunListener、CommandLineRunner、ApplicationRunner