**Springboot笔记3**

# 消息队列

## 介绍

1、什么是JMS: Java消息服务（Java Message Service),Java平台中关于面向消息中间件的接口

2、JMS是一种与厂商无关的 API，用来访问消息收发系统消息，它类似于JDBC(Java Database Connectivity)。这里，JDBC 是可以用来访问许多不同关系数据库的 API

3、使用场景：

1）跨平台

2）多语言

3）多项目

4）解耦

5）分布式事务

6）流量控制

7）最终一致性

8）RPC调用

上下游对接，数据源变动->通知下属

4、概念

JMS提供者：Apache ActiveMQ、RabbitMQ、Kafka、Notify、MetaQ、RocketMQ

JMS生产者(Message Producer)

JMS消费者(Message Consumer)

JMS消息

JMS队列

JMS主题

JMS消息通常有两种类型：点对点（Point-to-Point)、发布/订阅（Publish/Subscribe）

5、编程模型

MQ中需要用的一些类

ConnectionFactory ：连接工厂，JMS 用它创建连接

Connection ：JMS 客户端到JMS Provider 的连接

Session： 一个发送或接收消息的线程

Destination ：消息的目的地;消息发送给谁.

MessageConsumer / MessageProducer： 消息接收者，消费者

## ActiveMQ5.x

使用docker简单安装使用！

Name：队列名称。

Number Of Pending Messages：等待消费的消息个数。

Number Of Consumers：当前连接的消费者数目

Messages Enqueued：进入队列的消息总个数，包括出队列的和待消费的，这个数量只增不减。

Messages Dequeued：已经消费的消息数量。

步骤1：

<!-- 整合消息队列ActiveMQ -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-activemq</artifactId>

</dependency>

<!-- 如果配置线程池则加入 -->

<dependency>

<groupId>org.apache.activemq</groupId>

<artifactId>activemq-pool</artifactId>

</dependency>

步骤2：

#整合jms测试，安装在别的机器，防火墙和端口号记得开放

spring.activemq.broker-url=tcp://47.100.115.31:61616

#集群配置

#spring.activemq.broker-url=failover:(tcp://47.100.115.31:61616,tcp://47.100.115.31:61617)

spring.activemq.user=admin

spring.activemq.password=admin

#下列配置要增加依赖

spring.activemq.pool.enabled=true

spring.activemq.pool.max-connections=100

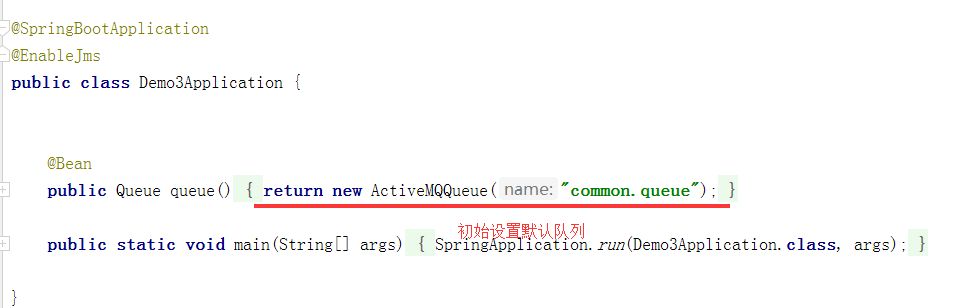
springboot启动类 @EnableJms，开启支持jms

@JmsListener(destination = "order.queue")

案例：

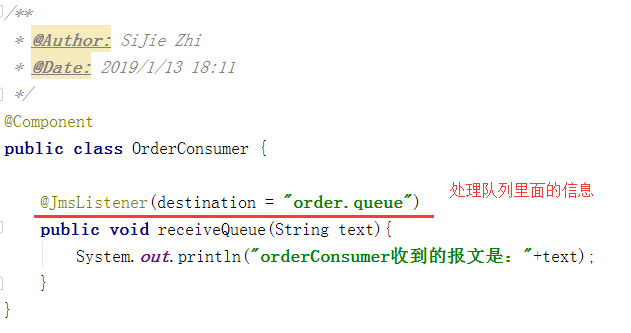


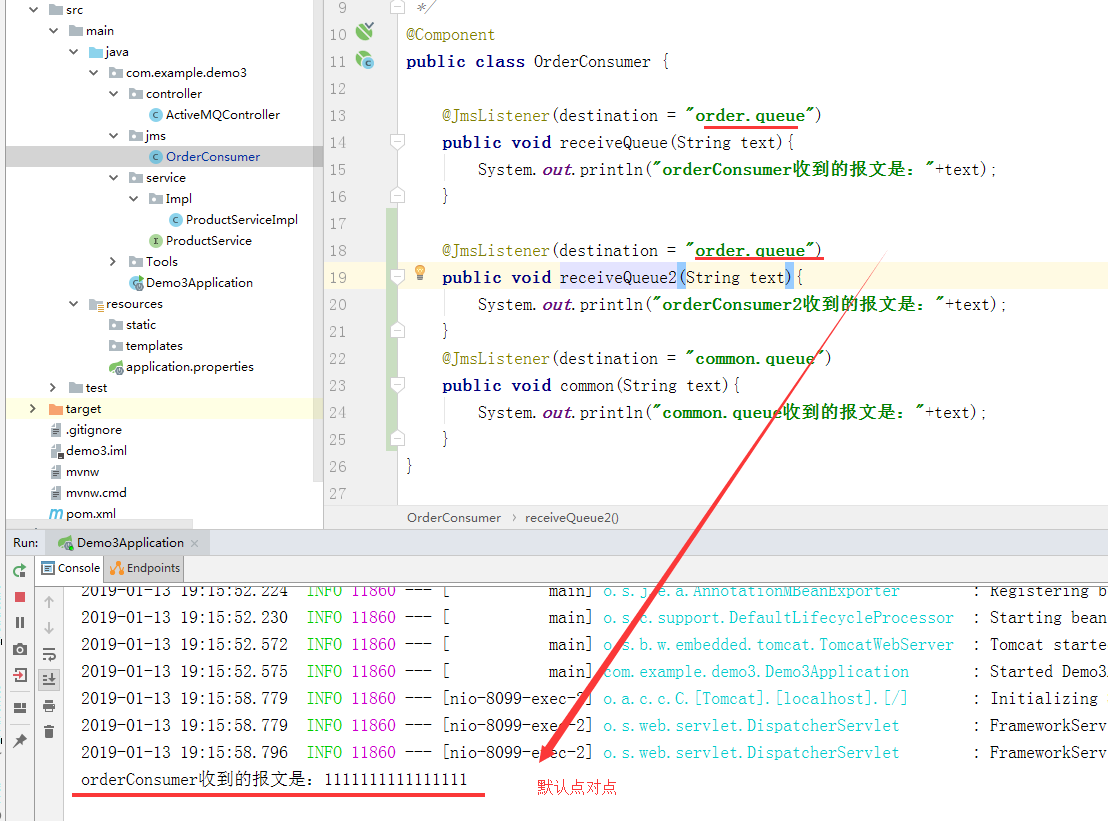












## ActiveMQ5订阅模式

默认是点对点

1、需要加入配置文件，支持发布订阅模型，默认只支持点对点

#default point to point

spring.jms.pub-sub-domain=true

注意点：

1、默认消费者并不会消费订阅发布类型的消息，这是由于springboot默认采用的是p2p模式进行消息的监听

修改配置：spring.jms.pub-sub-domain=true

2、@JmsListener如果不指定独立的containerFactory的话是只能消费queue消息

修改订阅者container：containerFactory="jmsListenerContainerTopic"

//需要给topic定义独立的JmsListenerContainer

@Bean

public JmsListenerContainerFactory<?> jmsListenerContainerTopic(ConnectionFactory activeMQConnectionFactory) {

DefaultJmsListenerContainerFactory bean = new DefaultJmsListenerContainerFactory();

bean.setPubSubDomain(true);

bean.setConnectionFactory(activeMQConnectionFactory);

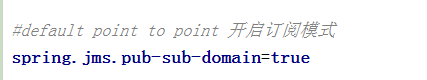
return bean;

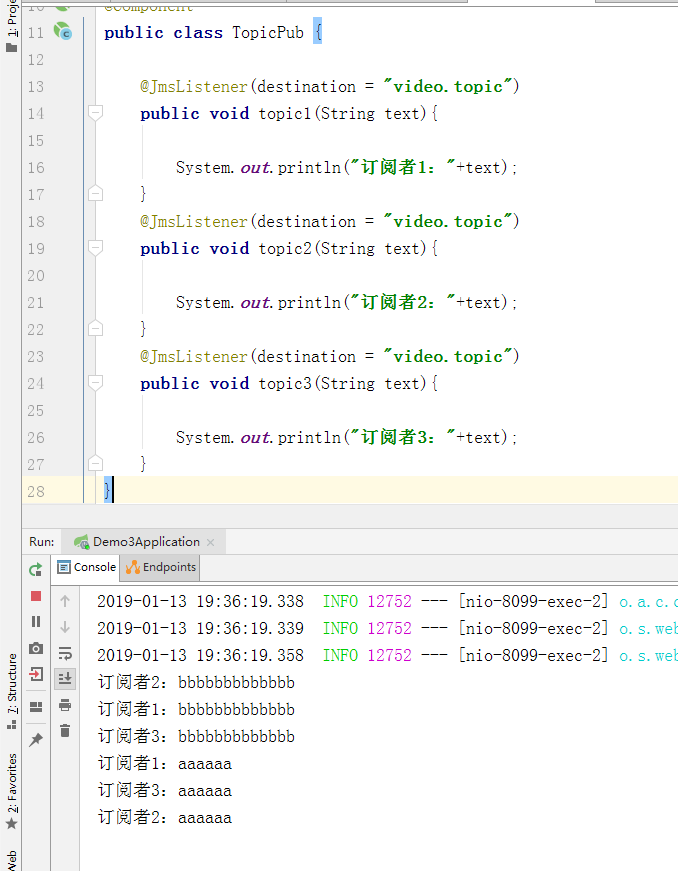
}

在配置文件里面，注释掉 #spring.jms.pub-sub-domain=true

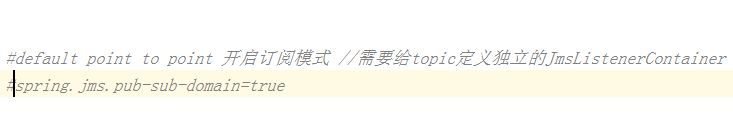
### 方式一：会只限制了只能用订阅模式

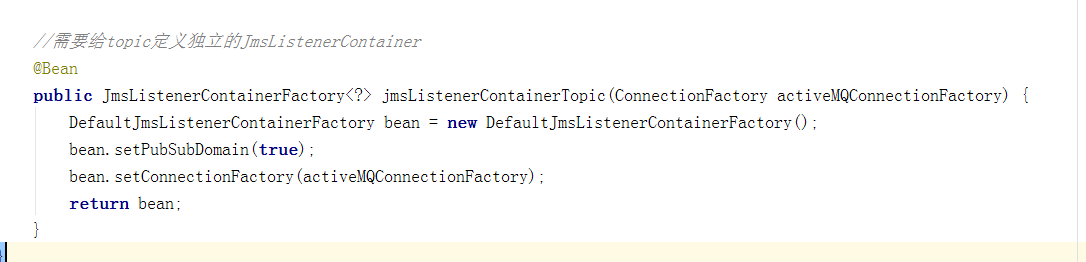


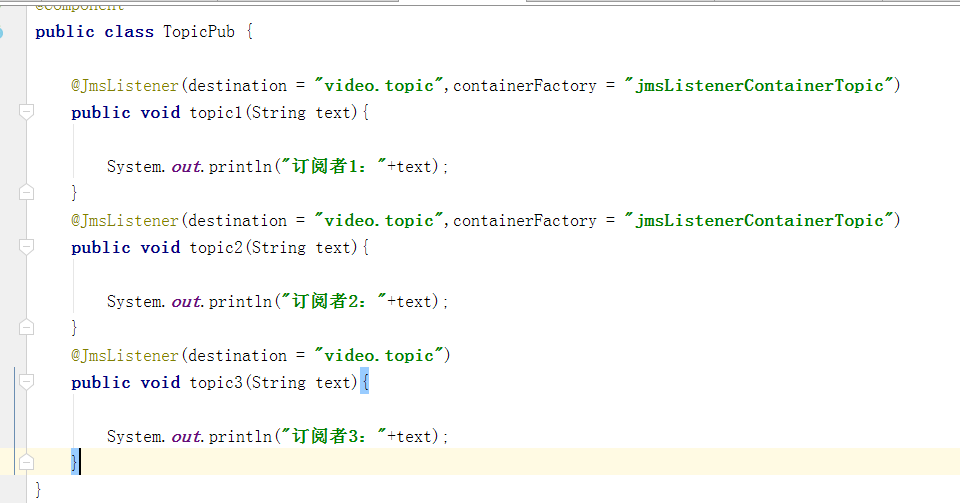




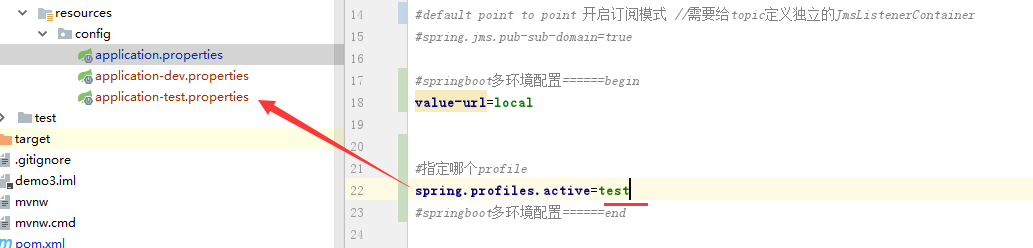
### 方式二：点对点和订阅连用







# 多环境配置



# 响应式编程Flux

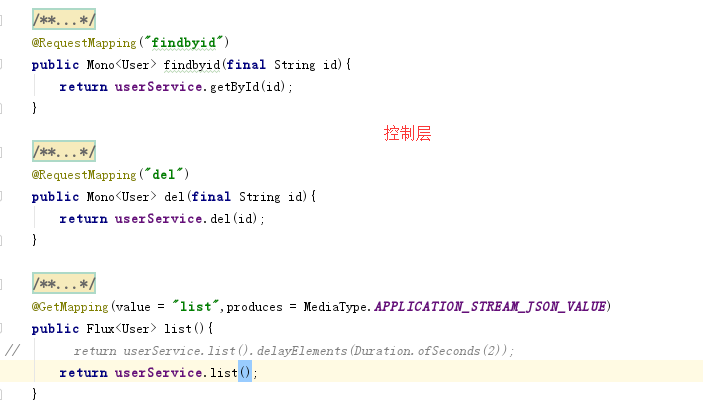
注意：

与Spring MVC不同，它不需要Servlet API，完全异步和非阻塞，并 通过Reactor项目实现Reactive Streams规范。

Spring WebFlux应用程序不严格依赖于Servlet API，因此它们不能作为war文件部署，也不能使用src/main/webapp目录







# 监控

## 导入依赖

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-actuator</artifactId>  
</dependency>

## 配置（可配可不配）

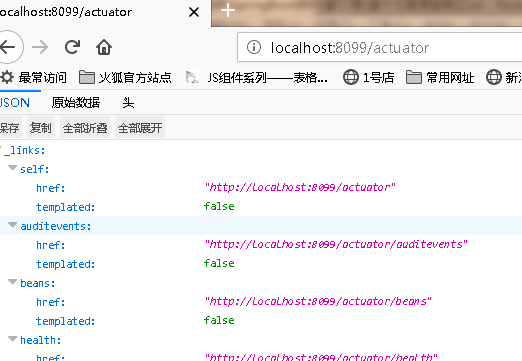
#开启全部

management.endpoints.web.exposure.include=\*

#management.endpoints.web.exposure.include=metrics

management.endpoints.web.exposure.exclude=env

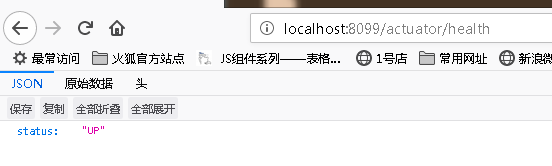
## 查看监控



## 使用场景

查看服务的健康状态。

定时调监控，如果服务有问题通知程序员维护。



正常是 up