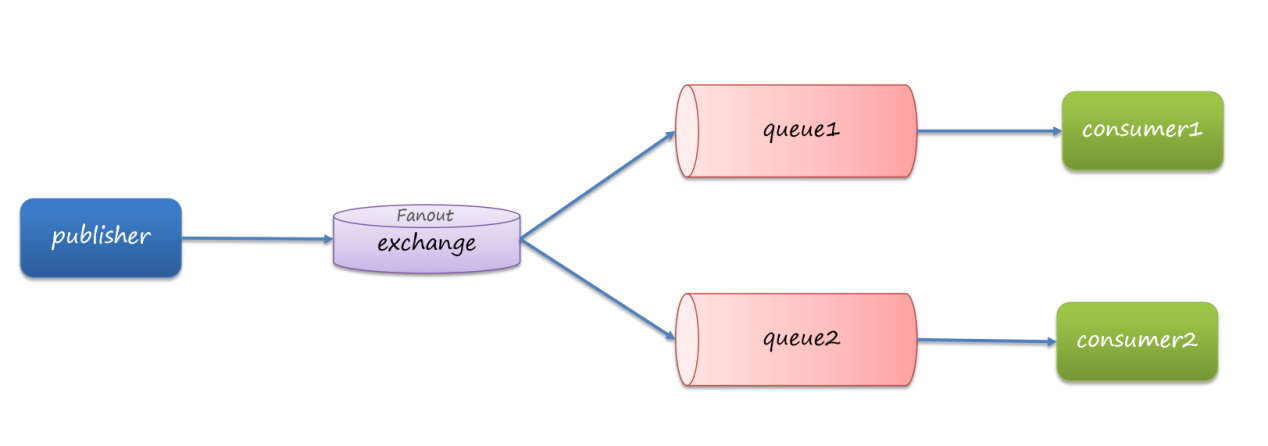
# SpringAMQP实现发布订阅-Fanout Exchange

我们来学习发布-订阅模式中的第一种交换机：Fanout Exchange

## Fanout Exchange广播交换机介绍

Fanout Exchange会将自己接收到的所有消息都路由到每一个跟其绑定的队列（Queue）！

你看你看你看，这个交换机不是完全符合我们前面说的需求吗：当用户支付完成了，你得去通知订单服务、仓储服务、短信服务、积分服务。。。。！



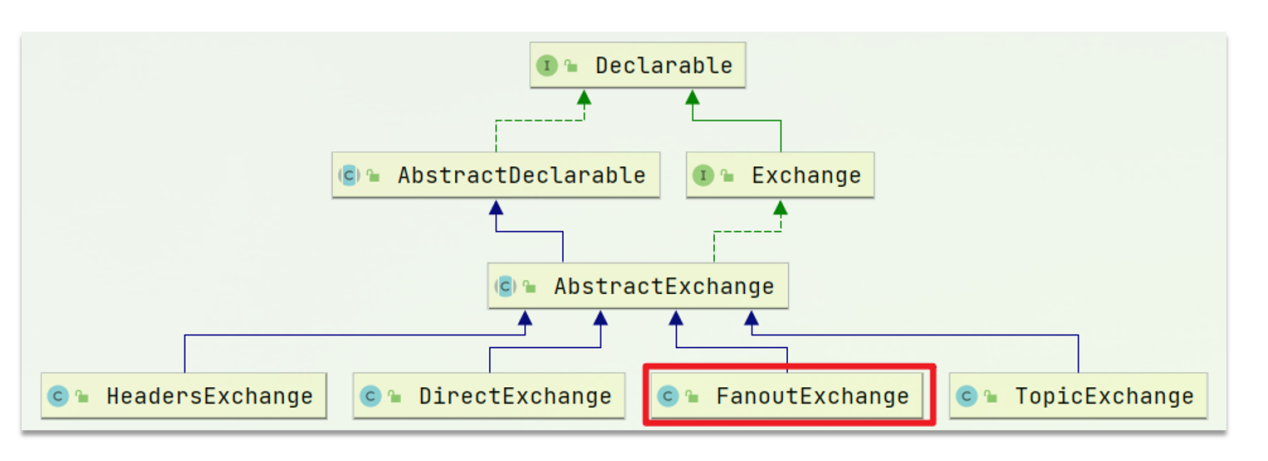
## 问题：如何实现队列和交换机绑定？

SpringAMQP提供了一系列的API，可以帮助我们实现声明交换机、声明队列、以及队列和交换机的绑定（API分别是：Exchange、Queue、Binding）。因此代码实现中，除了做常规的消息发送、消息的接收处理以外，我们还需要利用SpringAMQP提供的API去声明队列和交换机，并把它们绑定起来！

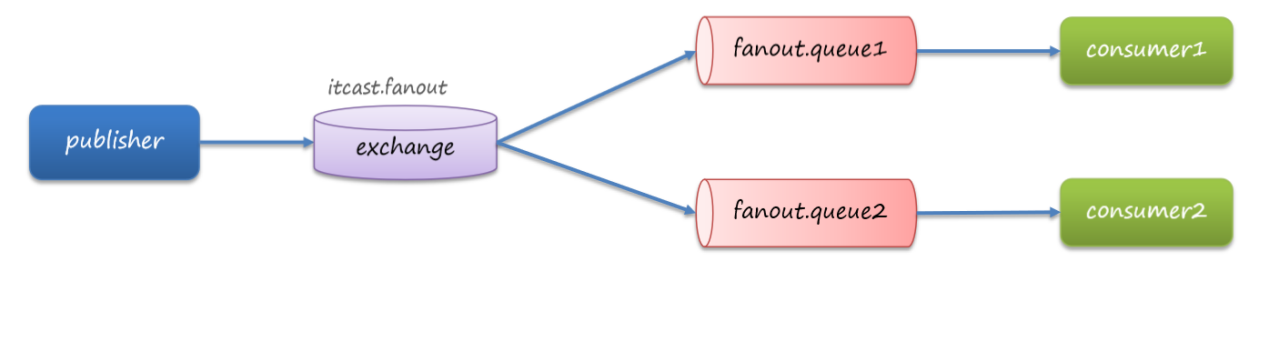
## 答案：使用SpringAMQP提供的API实现队列和交换机绑定

所以就是**使用SpringAMQP提供的API，完成声明交换机（Exchange）、声明队列（Queue）、以及队列和交换机的绑定（Binding）**

给你看一下SpringAMQP提供的API，你看你看，不是有声明各种交换机的各种类吗！至于声明队列的API就是Queue，声明交换机和队列绑定关系的API就叫Binding！



## 案例：使用SpringAMQP演示FanoutExchange的使用



### 1：在consumer服务中创建一个类，添加@Configuration注解，并声明FanoutExchange、Queue、和绑定关系对象Binding

在consumer服务中，利用SpriingAMQP的API声明队列（fanout.queue1和fanout.queue2）、交换机（hyh.fanout），并将两者绑定。

我们是通过声明@Bean的方式来写的，将来Spring读取到这些@Bean注解时，就会创建对应的Bean，并且会帮助我们到RabbitMQ去声明队列、交换机、绑定关系，全部都会自动帮我们做，我们不用手动去RabbitMQ管理后台操作，代码如下：

**import org.springframework.amqp.core.Binding;**

**import org.springframework.amqp.core.BindingBuilder;**

**import org.springframework.amqp.core.FanoutExchange;**

**import org.springframework.amqp.core.Queue;**

**import org.springframework.context.annotation.Bean;**

**import org.springframework.context.annotation.Configuration;**

**@Configuration**

**public class FanoutConfig {**

**@Bean**

**public FanoutExchange fanoutExchange() {**

**// 声明FanoutExchange广播交换机：hyh.fanout**

**return new FanoutExchange("hyh.fanout");**

**}**

**@Bean**

**public Queue fanoutQueue1() {**

**// 声明队列：fanout.queue1，注意方法名称就是bean的唯一ID**

**return new Queue("fanout.queue1");**

**}**

**@Bean**

**public Queue fanoutQueue2() {**

**// 声明队列：fanout.queue1**

**return new Queue("fanout.queue2");**

**}**

**@Bean**

**public Binding fanoutBinding1(FanoutExchange fanoutExchange,**

**Queue fanoutQueue1) {**

**// 绑定队列fanoutQueue1到交换机fanoutExchange**

**return BindingBuilder.bind(fanoutQueue1).to(fanoutExchange);**

**}**

**@Bean**

**public Binding fanoutBinding2(FanoutExchange fanoutExchange,**

**Queue fanoutQueue2) {**

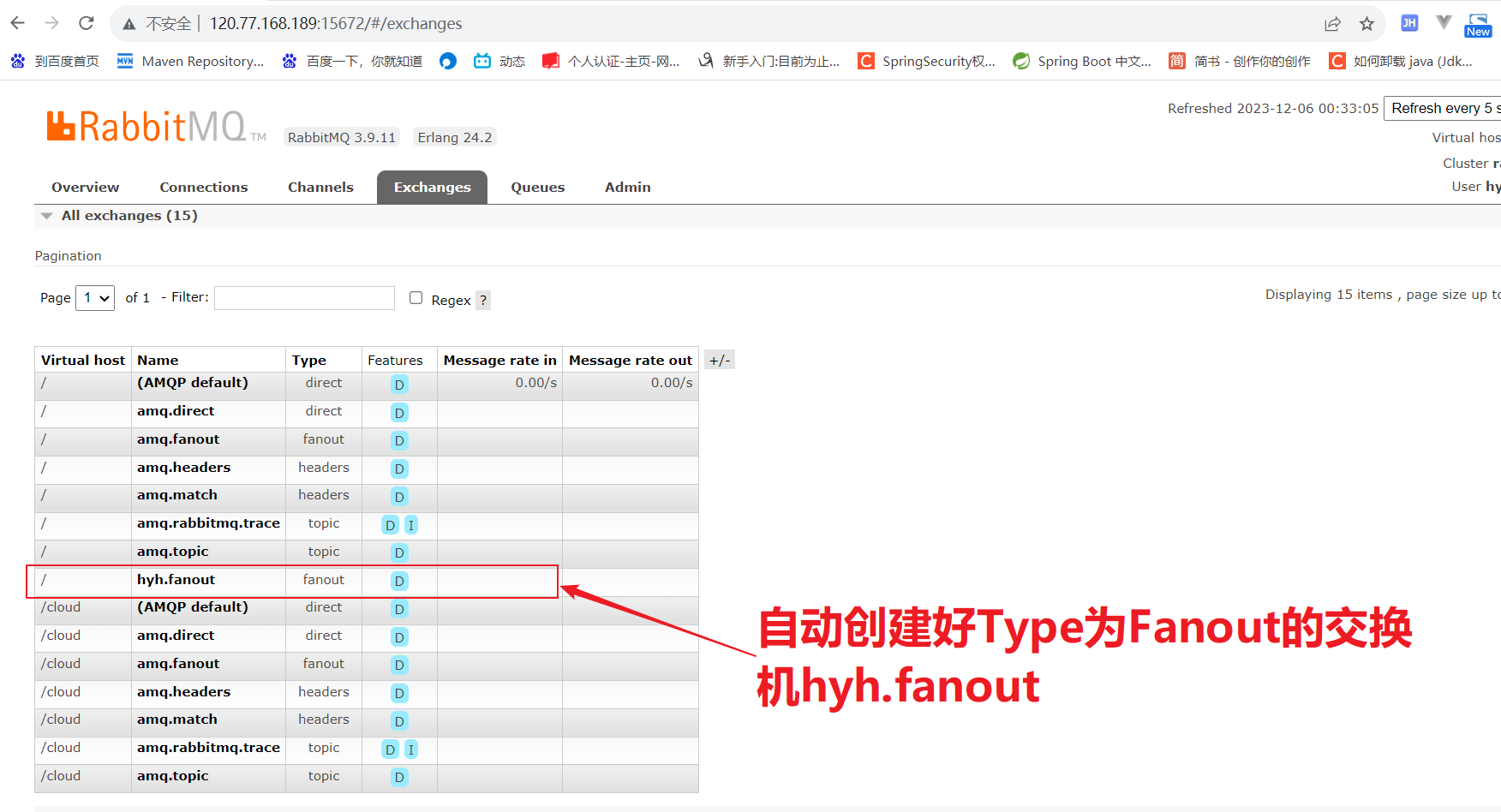
**// 绑定队列fanoutQueue2到交换机fanoutExchange**

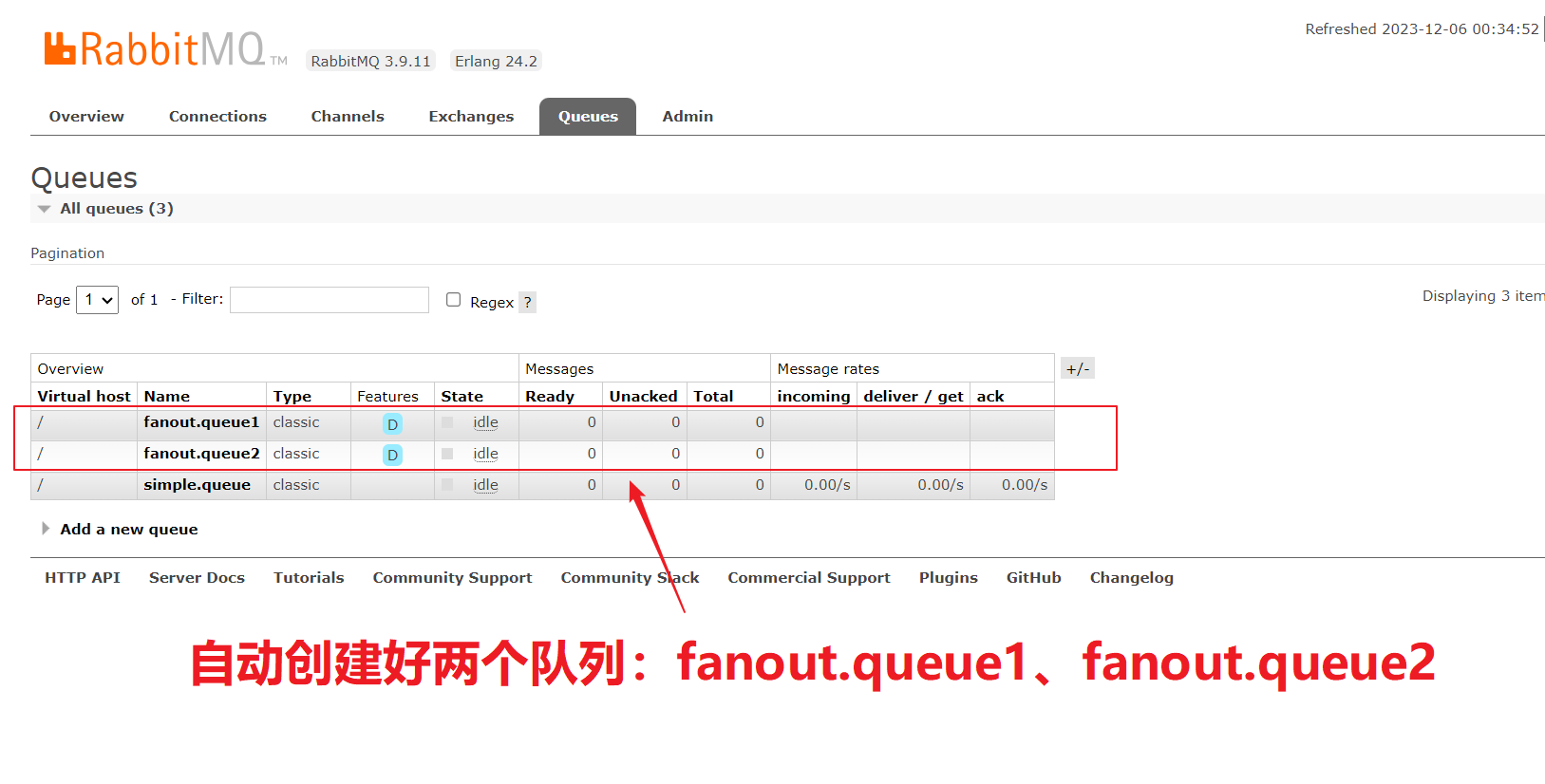
**return BindingBuilder.bind(fanoutQueue2).to(fanoutExchange);**

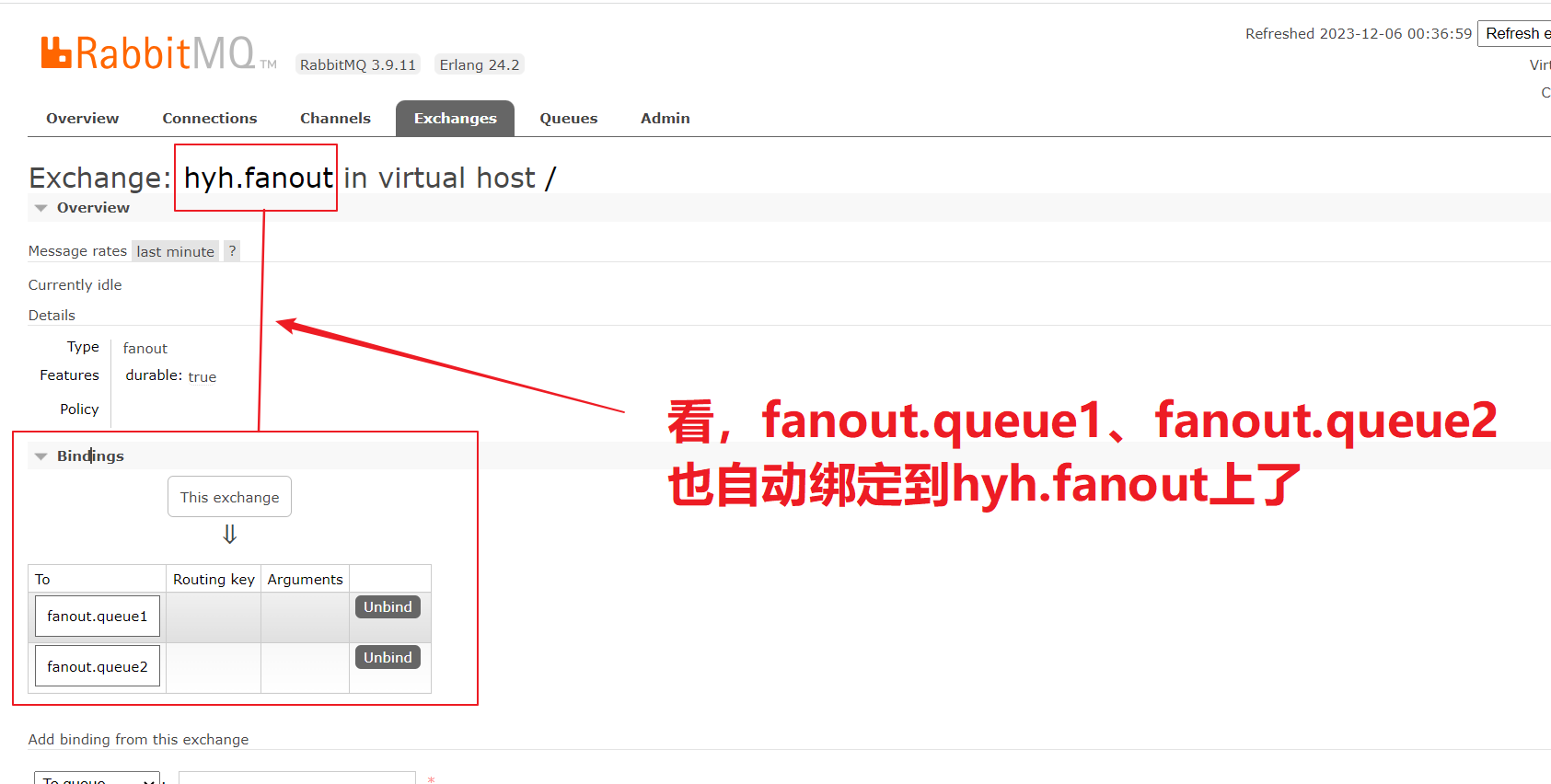
**}**

**}**

### 2：重启consumer，到RabbitMQ管理后台看一下是不是Spring会帮我们到RabbitMQ创建队列、交换机、绑定关系

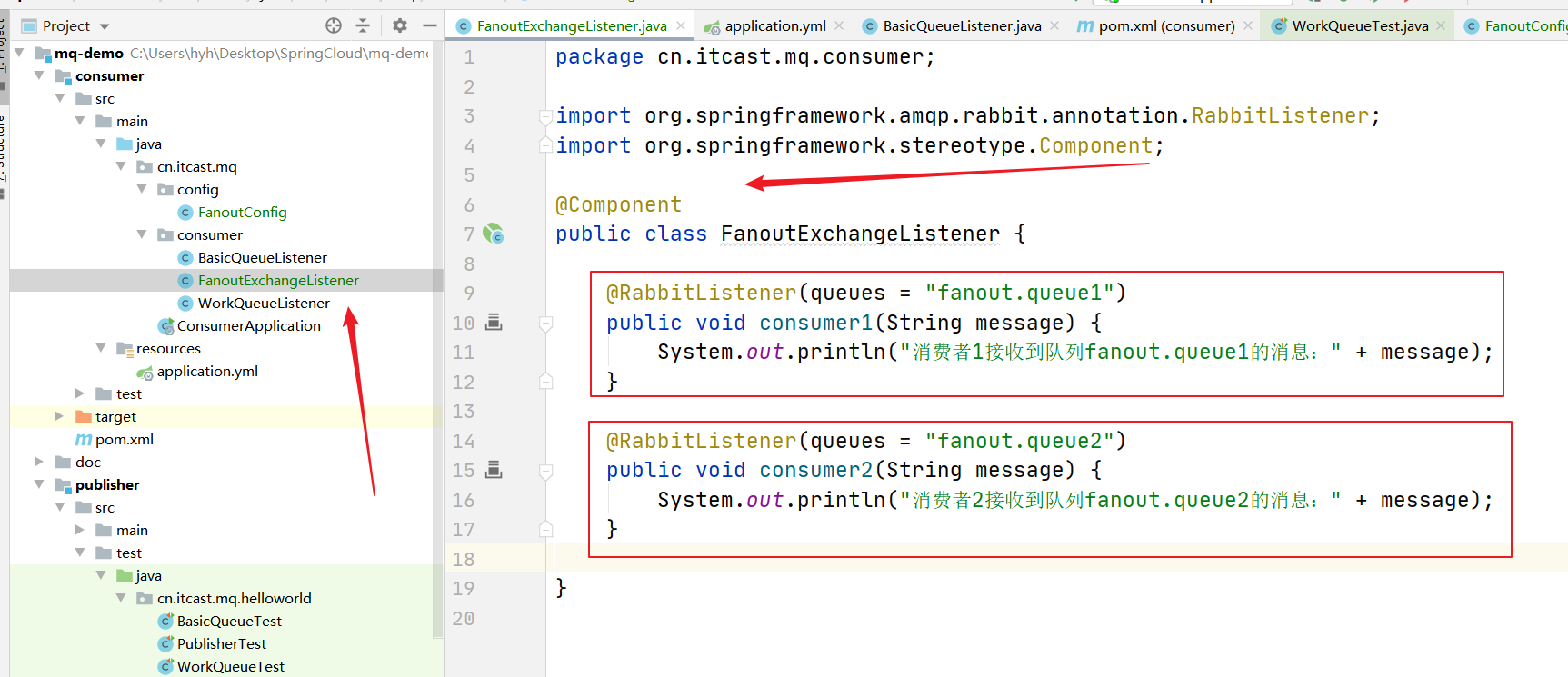






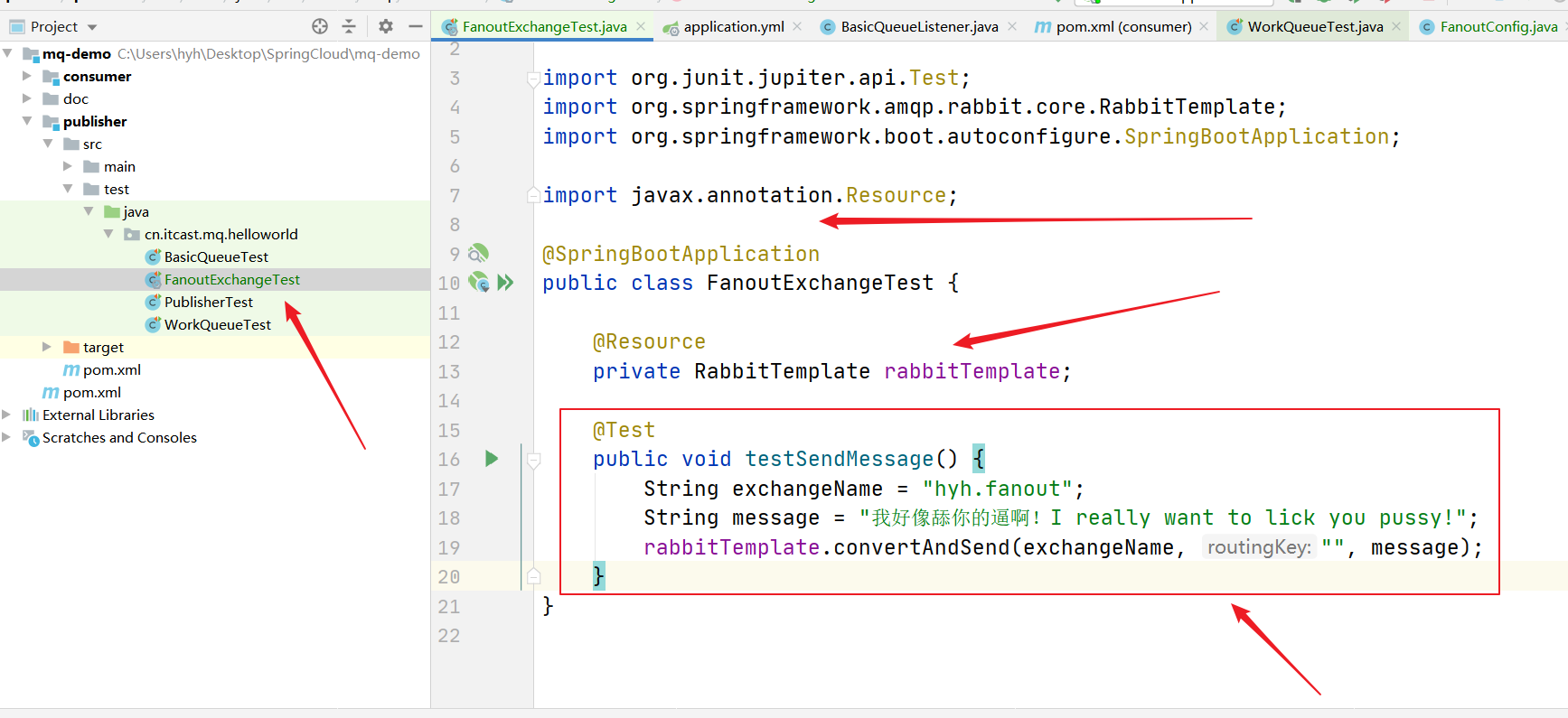
### 3：在consumer服务中，编写两个消费者方法，分别监听fanout.queue1和fanout.queue2;

编码完，重启consumer

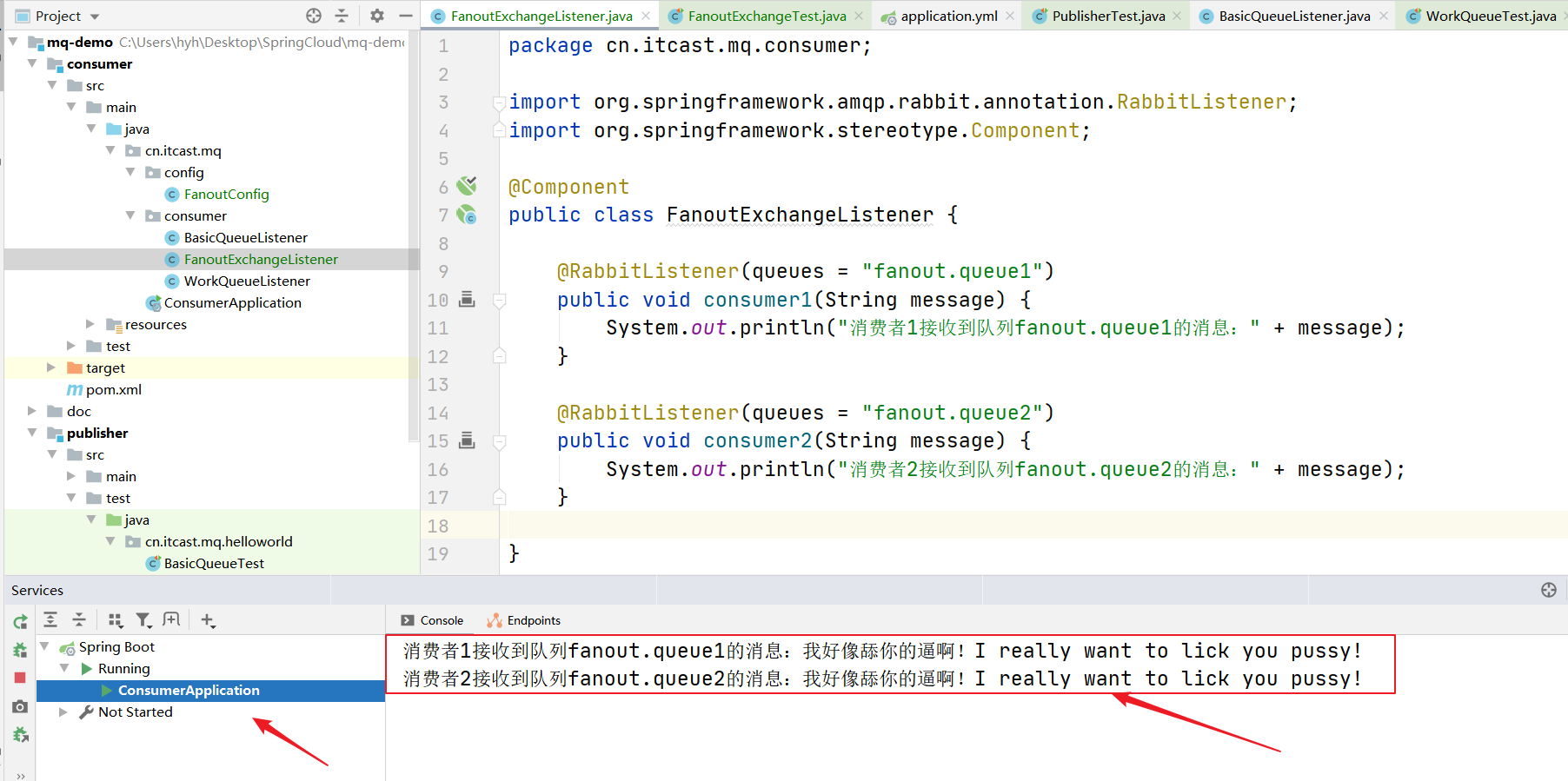


### 4：在publisher服务中编写测试方法，向hyh.fanout发送消息。

编码完，运行测试用例方法



### 5：在consumer中查看两个消费者是否都收到消息！



## 总结：

### 交换机的作用是什么？

1：接收publisher发送的消息；

2：将消息按照规则路由到与之绑定的队列

3：交换机不能缓存消息，路由失败，消息丢失；

4：FanoutExchange的交换机会将消息路由到每个绑定的队列

### 声明队列、交换机、绑定关系的Bean是什么？

Queue FanoutExchange  Binding

## 注意点：

### 1：交换机只是负责将消息投递到队列；

### 2：把消息投递到绑定一个队列，还是投递到绑定多个队列，是由交换机的类型决定的；

### 3：Fanout Exchange广播交换机会把消息投递到绑定的每个队列

### 4：交换机exchange和消费者consumer无关；

### 5：队列只有一个消费者监听，那么队列中的消息只有一个消费者接收；

### 6：队列有多个消费者监听，那么队列中的一个消息只会由一个消费者接收

你看懂了以上的注意点，就能明白为什么Fanout Exchange广播交换机适合我们之前提出的需求：当用户支付完成了，你得去通知订单服务、仓储服务、短信服务、积分服务。。。。！