# 步骤

### 生产者

- 3. 编写yml配置,基本信息配置
- 4. 定义交换机,队列以及绑定关系的配置类
- 5. 注入RabbitTemplate,调用方法,完成消息发送

### 消费者

- 3. 编写yml配置,基本信息配置
- 4. 定义监听类,使用@RabbitListener注解完成队列监听。

## 小结

- SpringBoot提供了快速整合RabbitMo的方式
- 基本信息再yml中配置,队列交互机以及绑定关系在配置类中使用Bean的方式配置
- 生产端直接注入RabbitTemplate完成消息发送
- 消费端直接使用@RabbitListener完成消息接收

```
@Bean
  public Queue duanxinQueue (
      return new Queue ( name:
  @Bean
  public Queue emailQueue() {
      return new Queue ( name: "email.fanout.queue", durable: true);
  // 3: 完成绑定关系(队列和交换机完成绑定关系)
@Bean
  public Binding smsBingding() {
      return BindingBuilder.bind(smsQueue()).to(fanoutExchange());
  @Bean
  public Binding emailBingding() {
      return BindingBuilder.bind(emailQueue()).to(fanoutExchange());
  @Bean
  public Binding duanxinBingding() {
      return BindingBuilder.bind(duanxinQueue()).to(fanoutExchange());
```

#### 接受消息

```
🖿 com ) 🖿 xuexiangban ) 🖿 rabbitmq ) 🖿 springbootorderrabbitmqconsumber ) 🖿 service ) 🖿 fanout ) 🌚 fanout ) 🕲 FanoutEmailConsumer 🕒 🔨 🐠 SpringbootOrderRabbitmqProducerApplicationTests context(cate
                             😌 😤 🗢 🦟 application.yml 🗵 📵 FanoutEmailConsumer.java 🗴 🕲 FanoutSMSConsumer.java 🗴 🕲 FanoutDuanxinConsumer.java
 ▼ springboot-order-rabbitmq-consumber D:\xuexian
   > III.mvn
                                                     import org.springframework.amqp.rabbit.annotation.RabbitHandler;

✓ I src

                                                     import org.springframework.amqp.rabbit.annotation.RabbitListener;

✓ Imain

→ java

          v 🖿 com.xuexiangban.rabbitmq.springbootor
                                                      * @description:

✓ Is service.fanout

                                                         @author: xuke
@time: 2021/3/6 0:38
                  EanoutDuanvinConsumer
                  FanoutEmailConsumer
                                                     @RabbitListener(queues = {"email.fanout.queue"})
               Springboot Order Rabbitmq Consumber
                                                     public class FanoutEmailConsumer {
                                                          @RabbitHandler
             static
                                                          public void reviceMessage(String message) {
            templates
                                                              System.out.println("email fanout---接收到了订单信息是: ->" + message);
             application.yml
      > test
```

### 消费端的交换机和队列绑定,使用注解的方式。推荐使用配置类

```
com.xuexiangban.rabbitmq.springbo

✓ □ config

    DirectRabbitMqConfiguration

                                                                                                                                                                    @description:

→ Service

                                                                                                                                                                      @author: xuke

✓ Image direct

                              © DirectDuanxinConsumer
                              © DirectEmailConsumer
                               © DirectSMSConsumer

→ Image of the property o
                              FanoutDuanxinConsumer
                               © FanoutEmailConsumer
                                                                                                                                                                                            key = "*.email.#"
                              EanoutSMSConsumer

✓ Image: topic

                              © TopicDuanxinConsume
                                                                                                                                                     public class TopicEmailConsumer {
                              © TopicEmailConsumer
                               TopicSMSConsumer
          Springboot Order Rabbitmq Consu
                                                                                                                                                                        public void reviceMessage(String message) { System.out.println("sms direct---接收到了订单信息是: ->" +
resources
templates
```

## producer-springboot

### RabbitMQConfig

```
package com.itheima.rabbitmq.config;
import org.springframework.amqp.core.*;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
@Configuration
public class RabbitMQConfig {
   public static final String EXCHANGE_NAME = "boot_topic_exchange";
   public static final String QUEUE_NAME = "boot_queue";
    //1.交换机
   @Bean("bootExchange")
   public Exchange bootExchange(){
       return ExchangeBuilder.topicExchange(EXCHANGE NAME).durable(true).build();
    }
   //2.Queue 队列
   @Bean("bootQueue")
   public Queue bootQueue(){
       return QueueBuilder.durable(QUEUE NAME).build();
    //3. 队列和交互机绑定关系 Binding
       1. 知道哪个队列
       2. 知道哪个交换机
       3. routing key
    */
   public Binding bindQueueExchange(@Qualifier("bootQueue") Queue queue,
@Qualifier("bootExchange") Exchange exchange){
       return BindingBuilder.bind(queue).to(exchange).with("boot.#").noargs();
   }
```

### application.yml

```
# 配置RabbitMQ的基本信息 ip 端口 username password..

spring:
    rabbitmq:
    host: 172.16.98.133 # ip
    port: 5672
    username: guest
    password: guest
    virtual-host: /
```

#### **ProducerTest**

```
package com.itheima.test;
import com.itheima.rabbitmq.config.RabbitMQConfig;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.amqp.rabbit.core.RabbitTemplate;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.test.context.junit4.SpringRunner;
@SpringBootTest
@RunWith(SpringRunner.class)
public class ProducerTest {
   //1.注入RabbitTemplate
    @Autowired
   private RabbitTemplate rabbitTemplate;
    @Test
   public void testSend(){
        rabbitTemplate.convertAndSend(RabbitMQConfig.EXCHANGE_NAME, "boot.haha", "boot mq
hello~~~");
   }
}
```

### consumer-springboot

```
spring:
rabbitmq:
host: 172.16.98.133 #主机ip
port: 5672 #端口
username: guest
password: guest
virtual-host: /
```

### RabbimtMQListener

```
@SpringBootApplication
public class ConsumerSpringbootApplication {
   public static void main(String[] args) {
        SpringApplication.run(ConsumerSpringbootApplication.class, args);
   }
}
```