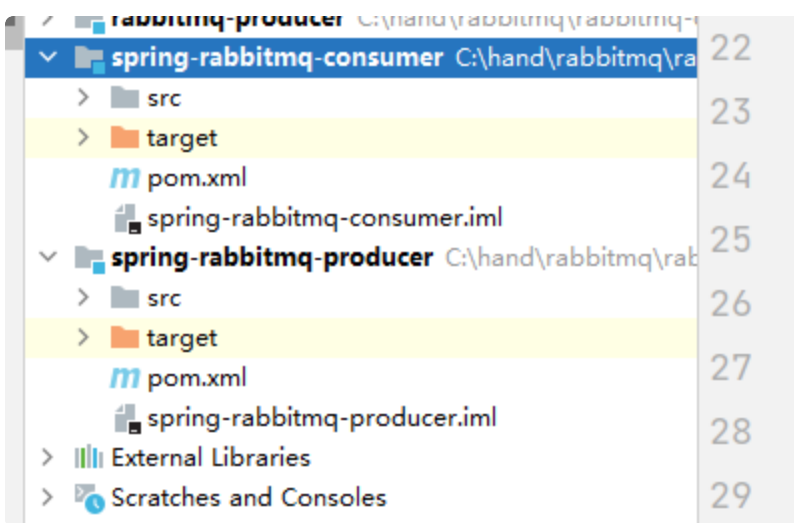
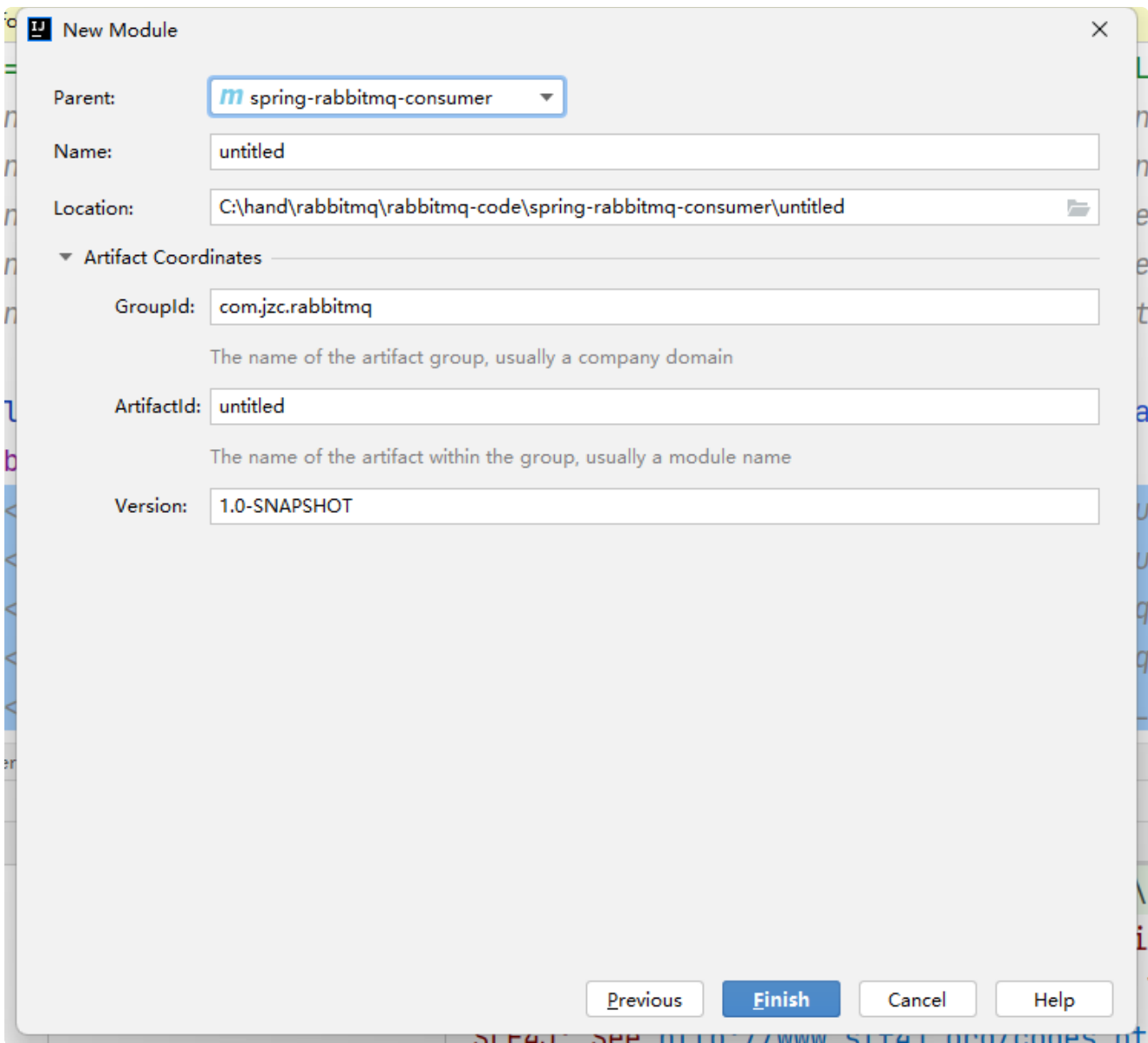


# Spring整合RabbitMQ

---

- 1.创建好消费者模块和生产者模块
- 2.分别配置好pom文件
- 3.分别设置RabbitMQ服务器基本配置
- 4.配置Producer的xml文件
- 5.配置Consumer的xml文件
- 6.创建Producer测试类发送各种消息
- 7.创建Ponsumer监听类接收队列消息

## 1.创建好消费者模块和生产者模块



## 2. 分别配置好pom文件

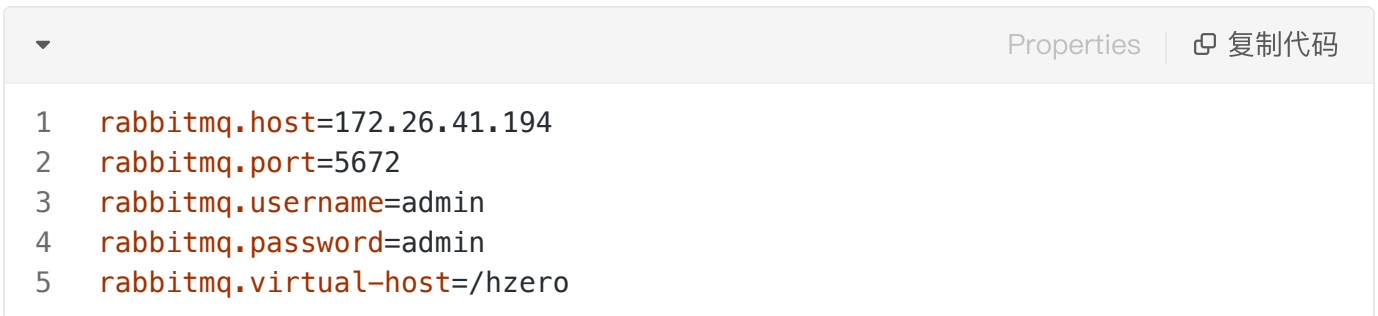
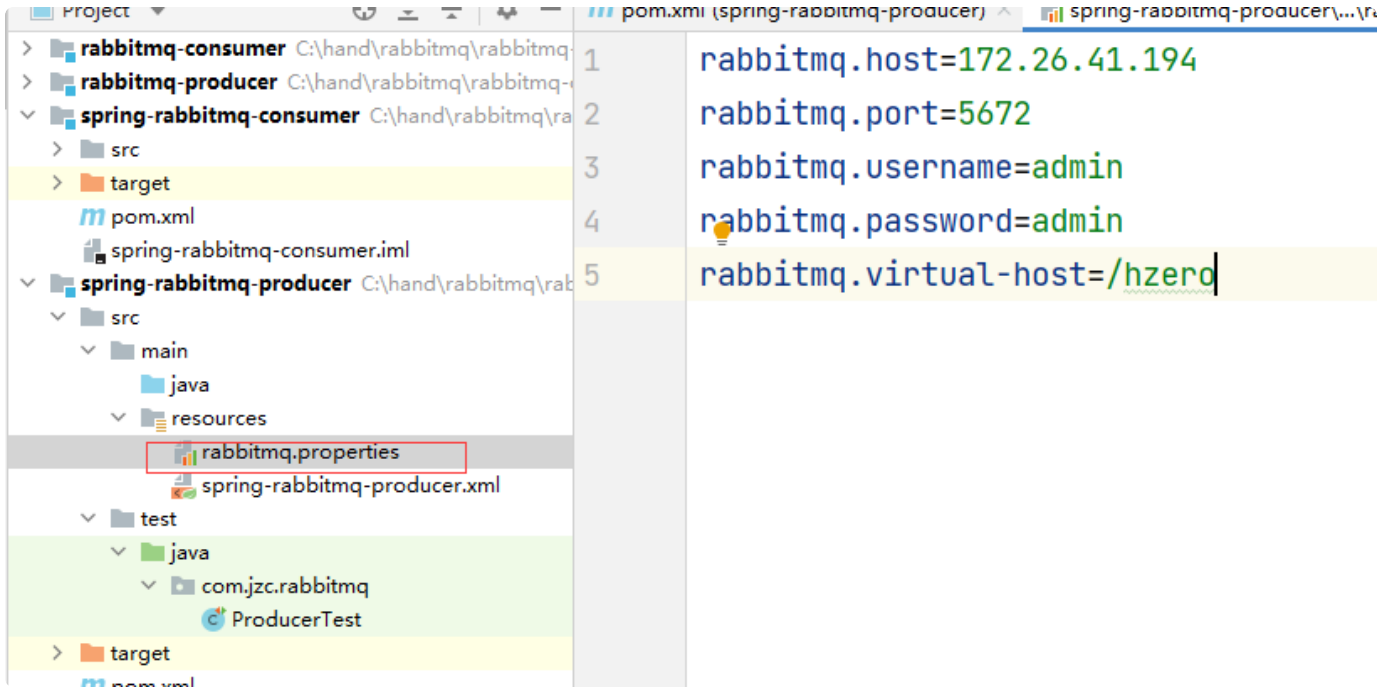
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5   <modelVersion>4.0.0</modelVersion>
6
7   <groupId>com.jzc.rabbitmq</groupId>
8   <artifactId>spring-rabbitmq-producer</artifactId>
9   <version>1.0-SNAPSHOT</version>
10
11   <dependencies>
12     <dependency>
13       <groupId>org.springframework</groupId>
14       <artifactId>spring-context</artifactId>
15       <version>5.1.7.RELEASE</version>
16     </dependency>
17
18     <dependency>
19       <groupId>org.springframework.amqp</groupId>
20       <artifactId>spring-rabbit</artifactId>
21       <version>2.1.8.RELEASE</version>
22     </dependency>
23
24     <dependency>
25       <groupId>junit</groupId>
26       <artifactId>junit</artifactId>
27       <version>4.12</version>
28     </dependency>
29
30     <dependency>
31       <groupId>org.springframework</groupId>
32       <artifactId>spring-test</artifactId>
33       <version>5.1.7.RELEASE</version>
34     </dependency>
35   </dependencies>
36
37   <build>
38     <plugins>
39       <plugin>
40         <groupId>org.apache.maven.plugins</groupId>
41         <artifactId>maven-compiler-plugin</artifactId>
42         <version>3.8.0</version>
43         <configuration>
44           <source>1.8</source>
```

```

45         <target>1.8</target>
46     </configuration>
47 </plugin>
48 </plugins>
49 </build>
50
51 </project>

```

### 3.分别设置RabbitMQ服务器基本配置



### 4.配置Producer的xml文件

The screenshot shows an IDE with the following components:

- Project Explorer:** Shows the project structure. The 'spring-rabbitmq-producer.xml' file is highlighted in the 'resources' directory.
- Editor:** Displays the XML content of 'spring-rabbitmq-producer.xml'. The content is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:rabbit="http://www.springframework.org/schema/rabbit"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans.xsd
                           http://www.springframework.org/schema/context
                           http://www.springframework.org/schema/context/spring-context.xsd
                           http://www.springframework.org/schema/rabbit
                           http://www.springframework.org/schema/rabbit/spring-rabbit.xsd">
  <!-- 加载配置文件 -->
  <context:property-placeholder location="classpath:rabbitmq.properties"/>

  <!-- 定义rabbitmq connectionFactory -->
  <rabbit:connection-factory id="connectionFactory" host="172.26.41.194"
                           port="5672"
                           username="admin"
                           password="admin"
                           virtual-host="/hzero"/>

  <!-- 定义管理交换机、队列 -->
  <rabbit:admin connection-factory="connectionFactory"/>
</beans>
```

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xmlns:context="http://www.springframework.org/schema/context"
5     xmlns:rabbit="http://www.springframework.org/schema/rabbit"
6     xsi:schemaLocation="http://www.springframework.org/schema/beans
7         http://www.springframework.org/schema/beans/spring-beans.xsd
8         http://www.springframework.org/schema/context
9         https://www.springframework.org/schema/context/spring-context.xsd
10        http://www.springframework.org/schema/rabbit
11        http://www.springframework.org/schema/rabbit/spring-rabbit.xsd">
12     <!--加载配置文件-->
13     <context:property-placeholder location="classpath:rabbitmq.properties"/>
14
15     <!-- 定义rabbitmq connectionFactory -->
16     <rabbit:connection-factory id="connectionFactory" host="${rabbitmq.hos
17         t}"
18         port="${rabbitmq.port}"
19         username="${rabbitmq.username}"
20         password="${rabbitmq.password}"
21         virtual-host="${rabbitmq.virtual-host}"/>
22     <!--定义管理交换机、队列-->
23     <rabbit:admin connection-factory="connectionFactory"/>
24
25     <!--定义持久化队列，不存在则自动创建；不绑定到交换机则绑定到默认交换机
26     默认交换机类型为direct，名字为：""，路由键为队列的名称
27     -->
28     <rabbit:queue id="spring_queue" name="spring_queue" auto-declare="true"/
29     >
30
31     <!-- ~~~~~广播；所有队列都能收到消息~~~~~ -->
32     <!--定义广播交换机中的持久化队列，不存在则自动创建-->
33     <rabbit:queue id="spring_fanout_queue_1" name="spring_fanout_queue_1" au
34         to-declare="true"/>
35
36     <!--定义广播交换机中的持久化队列，不存在则自动创建-->
37     <rabbit:queue id="spring_fanout_queue_2" name="spring_fanout_queue_2" au
38         to-declare="true"/>
39
40     <!--定义广播类型交换机；并绑定上述两个队列-->
41     <rabbit:fanout-exchange id="spring_fanout_exchange" name="spring_fanout_
42         exchange" auto-declare="true">
43         <rabbit:bindings>
44             <rabbit:binding queue="spring_fanout_queue_1"/>

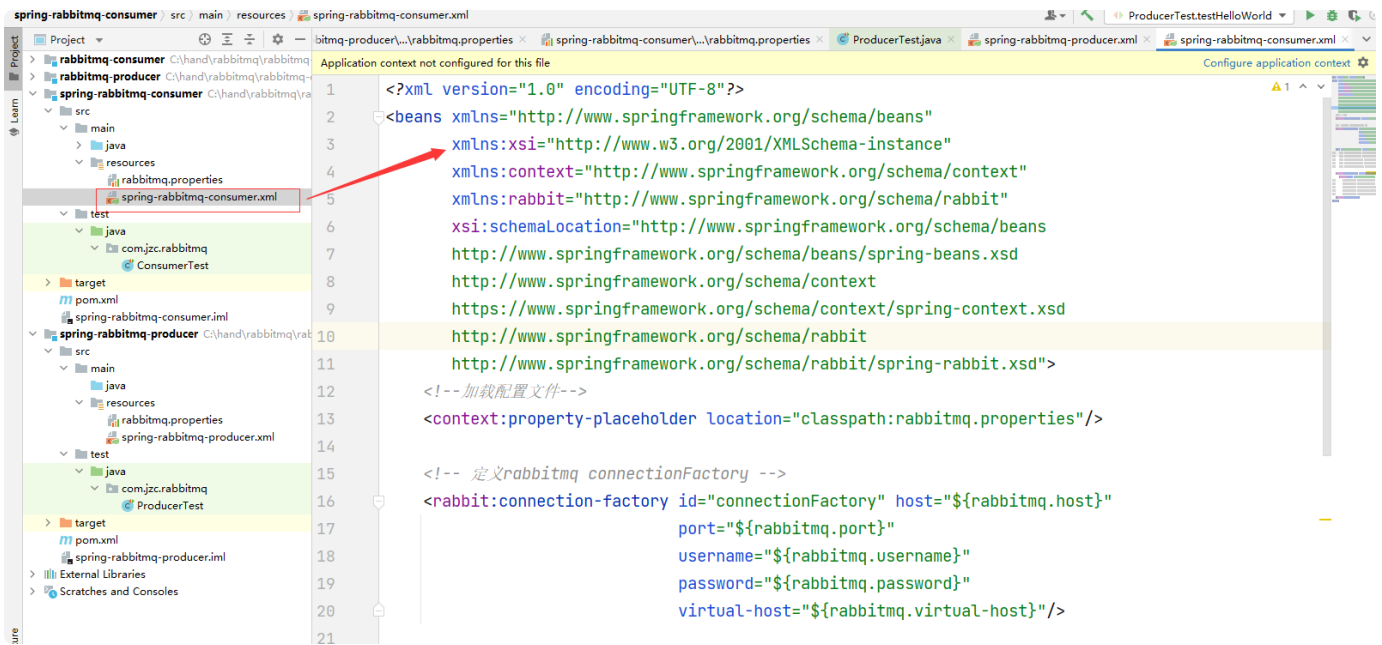
```

```

40         <rabbit:binding queue="spring_fanout_queue_2"/>
41     </rabbit:bindings>
42 </rabbit:fanout-exchange>
43
44     <!-- ~~~~~通配符；*匹配一个单词，#匹配多个单词 ~~~~~
~~~~~ -->
45     <!--定义广播交换机中的持久化队列，不存在则自动创建-->
46     <rabbit:queue id="spring_topic_queue_star" name="spring_topic_queue_star" auto-declare="true"/>
47     <!--定义广播交换机中的持久化队列，不存在则自动创建-->
48     <rabbit:queue id="spring_topic_queue_well" name="spring_topic_queue_well" auto-declare="true"/>
49     <!--定义广播交换机中的持久化队列，不存在则自动创建-->
50     <rabbit:queue id="spring_topic_queue_well2" name="spring_topic_queue_well2" auto-declare="true"/>
51
52     <rabbit:topic-exchange id="spring_topic_exchange" name="spring_topic_exchange" auto-declare="true">
53         <rabbit:bindings>
54             <rabbit:binding pattern="jzc.*" queue="spring_topic_queue_star"/>
55             <rabbit:binding pattern="jzc.#" queue="spring_topic_queue_well"/>
56             <rabbit:binding pattern="jzcHzero.#" queue="spring_topic_queue_well2"/>
57         </rabbit:bindings>
58     </rabbit:topic-exchange>
59
60     <!--定义rabbitTemplate对象操作可以在代码中方便发送消息-->
61     <rabbit:template id="rabbitTemplate" connection-factory="connectionFactory"/>
62 </beans>

```

## 5.配置Consumer的xml文件





```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xmlns:context="http://www.springframework.org/schema/context"
5      xmlns:rabbit="http://www.springframework.org/schema/rabbit"
6      xsi:schemaLocation="http://www.springframework.org/schema/beans
7      http://www.springframework.org/schema/beans/spring-beans.xsd
8      http://www.springframework.org/schema/context
9      https://www.springframework.org/schema/context/spring-context.xsd
10     http://www.springframework.org/schema/rabbit
11     http://www.springframework.org/schema/rabbit/spring-rabbit.xsd">
12     <!--加载配置文件-->
13     <context:property-placeholder location="classpath:rabbitmq.properties"
14 />
15     <!-- 定义rabbitmq connectionFactory -->
16     <rabbit:connection-factory id="connectionFactory" host="${rabbitmq.hos
17 t}"
18         port="${rabbitmq.port}"
19         username="${rabbitmq.username}"
20         password="${rabbitmq.password}"
21         virtual-host="${rabbitmq.virtual-host}"/>
22     <bean id="springQueueListener" class="com.jzc.rabbitmq.listener.Spring
23 QueueListener"/>
24     <!-- <bean id="fanoutListener1" class="com.jzc.rabbitmq.listener.Fanout
25 Listener1"/>-->
26     <!-- <bean id="fanoutListener2" class="com.jzc.rabbitmq.listener.Fanout
27 Listener2"/>-->
28     <!-- <bean id="topicListenerStar" class="com.jzc.rabbitmq.listener.Topi
29 cListenerStar"/>-->
30     <!-- <bean id="topicListenerWell" class="com.jzc.rabbitmq.listener.Topi
31 cListenerWell"/>-->
32     <!-- <bean id="topicListenerWell2" class="com.jzc.rabbitmq.listener.Topi
33 cListenerWell2"/>-->
34     <rabbit:listener-container connection-factory="connectionFactory" auto
35 -declare="true">
36         <rabbit:listener ref="springQueueListener" queue-names="spring_que
37 ue"/>
38         <!-- <rabbit:listener ref="fanoutListener1" queue-names="spring_fan
39 out_queue_1"/>-->
40         <!-- <rabbit:listener ref="fanoutListener2" queue-names="spring_fan
41 out_queue_2"/>-->

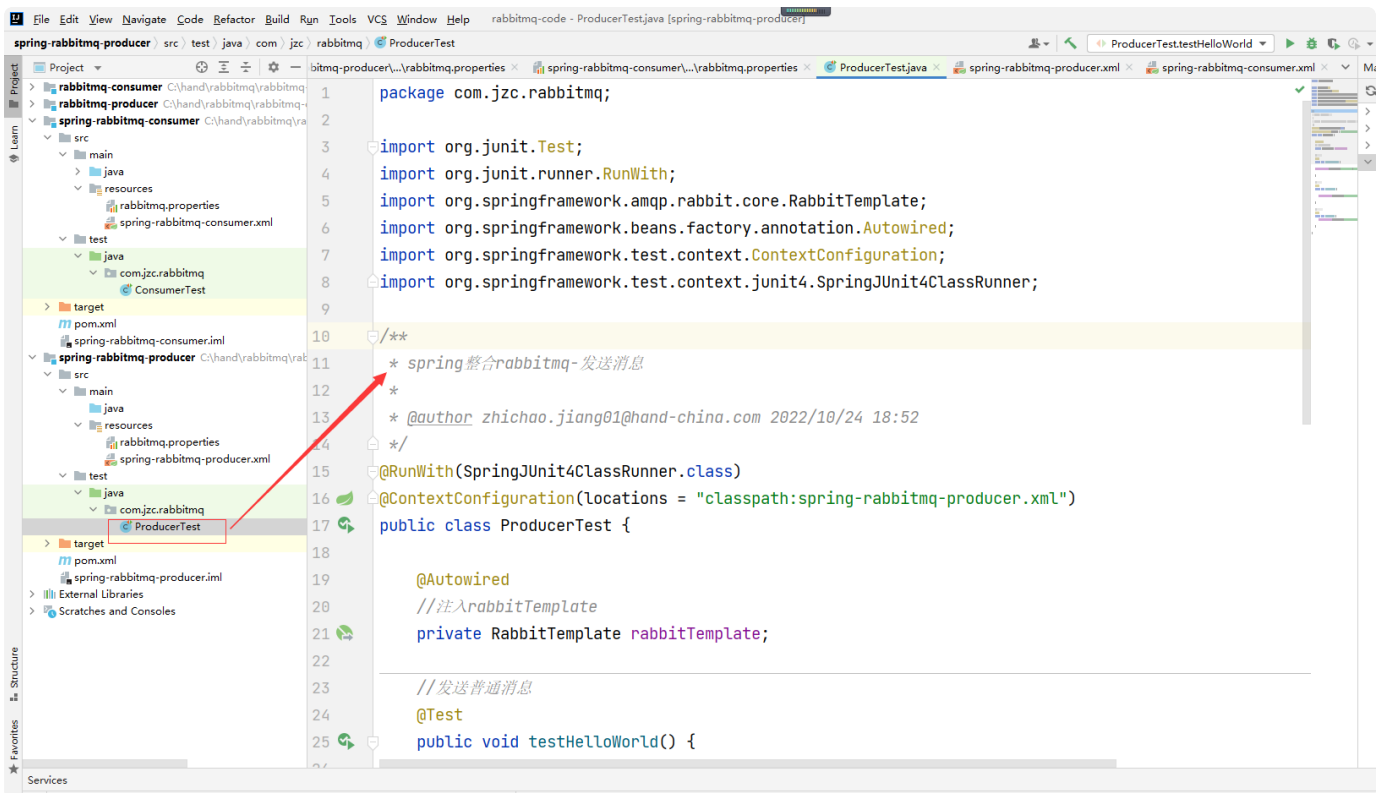
```

```

34 <!--      <rabbit:listener ref="topicListenerStar" queue-names="spring_t
opic_queue_star"/>-->
35 <!--      <rabbit:listener ref="topicListenerWell" queue-names="spring_t
opic_queue_well"/>-->
36 <!--      <rabbit:listener ref="topicListenerWell2" queue-names="spring_
topic_queue_well2"/>-->
37     </rabbit:listener-container>
</beans>

```

## 6.创建Producer测试类发送各种消息



```
1  /**
2   * spring整合rabbitmq-发送消息
3   *
4   * @author zhichao.jiang01@hand-china.com 2022/10/24 18:52
5   */
6   @RunWith(SpringJUnit4ClassRunner.class)
7   @ContextConfiguration(locations = "classpath:spring-rabbitmq-producer.
  xml")
8   public class ProducerTest {
9
10      @Autowired
11      //注入rabbitTemplate
12      private RabbitTemplate rabbitTemplate;
13
14      //发送普通消息
15      @Test
16      public void testHelloWorld() {
17
18          rabbitTemplate.convertAndSend("spring_queue","hello worl
  d.....");
19
20      }
21
22      //发送广播消息
23      @Test
24      public void testFanout() {
25
26          rabbitTemplate.convertAndSend("spring_fanout_exchange","", "hel
  lo fanout.....");
27
28      }
29
30      //发送通配符消息
31      @Test
32      public void testTopic() {
33          rabbitTemplate.convertAndSend("spring_topic_exchange","jzc.hh.
  ee","hello topic.....");
34
35      }
36
37  }
38
```

## Queues

▼ All queues (15)

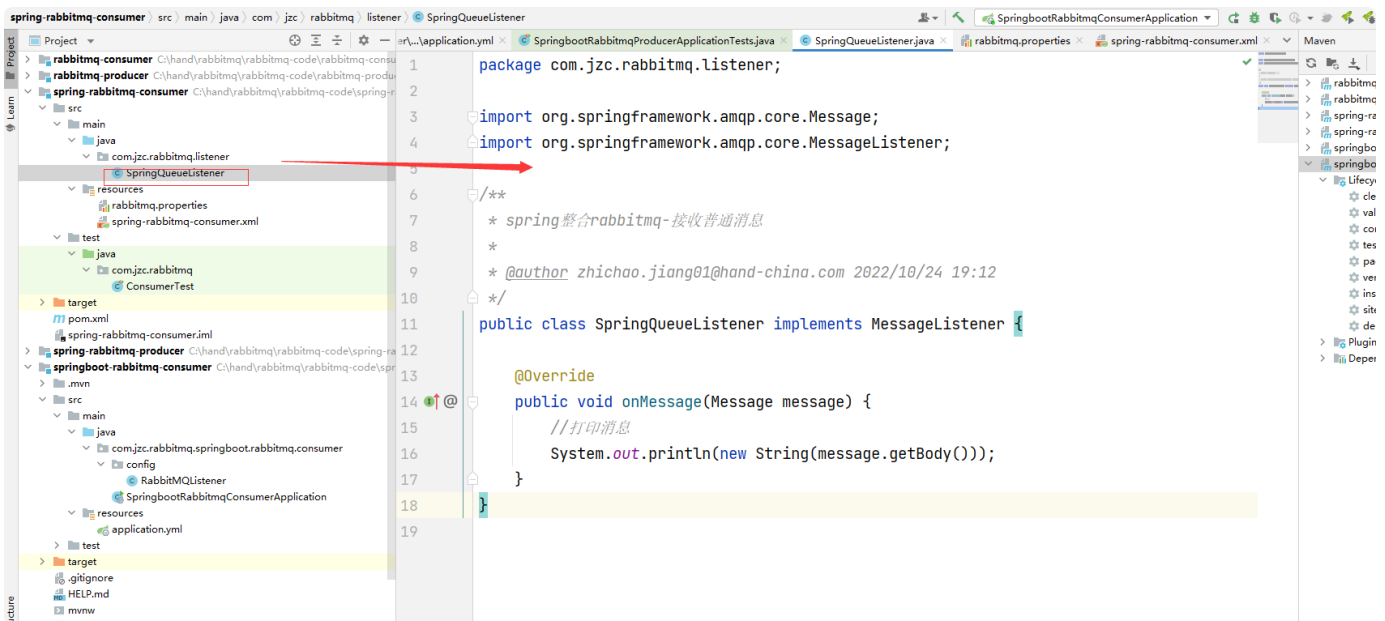
Pagination

Page 1 of 1 - Filter:  ☐ Regex ?

Overview					Messages			Message rates			+/-
Virtual host	Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack	
/hzero	boot_queue	classic	<span>D</span>	idle	1	0	1	0.00/s	0.00/s	0.00/s	
/hzero	hello_world	classic	<span>D</span>	idle	0	0	0				
/hzero	spring_fanout_queue_1	classic	<span>D</span>	idle	1	0	1	0.00/s			
/hzero	spring_fanout_queue_2	classic	<span>D</span>	idle	1	0	1	0.00/s			
/hzero	spring_queue	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	
/hzero	spring_topic_queue_star	classic	<span>D</span>	idle	0	0	0				
/hzero	spring_topic_queue_well	classic	<span>D</span>	idle	1	0	1	0.00/s			
/hzero	spring_topic_queue_well2	classic	<span>D</span>	idle	0	0	0				
/hzero	test_direct_queue1	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	
/hzero	test_direct_queue2	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	
/hzero	test_fanout_queue1	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	
/hzero	test_fanout_queue2	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	
/hzero	test_topic_queue1	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	
/hzero	test_topic_queue2	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	
/hzero	workQueue	classic	<span>D</span>	idle	0	0	0	0.00/s	0.00/s	0.00/s	

► Add a new queue

## 7.创建Ponsumer监听类接收队列消息



```

package com.jzc.rabbitmq.listener;

import org.springframework.amqp.core.Message;
import org.springframework.amqp.core.MessageListener;

/**
 * spring整合rabbitmq-接收普通消息
 *
 * @author zhichao.jiang01@hand-china.com 2022/10/24 19:12
 */
public class SpringQueueListener implements MessageListener {

    @Override
    public void onMessage(Message message) {
        //打印消息
        System.out.println(new String(message.getBody()));
    }
}

```

```

1  /**
2   * spring整合rabbitmq-接收普通消息
3   *
4   * @author zhichao.jiang01@hand-china.com 2022/10/24 19:12
5   */
6  public class SpringQueueListener implements MessageListener {
7
8      @Override
9      public void onMessage(Message message) {
10         //打印消息
11         System.out.println(new String(message.getBody()));
12     }
13 }
14

```

