

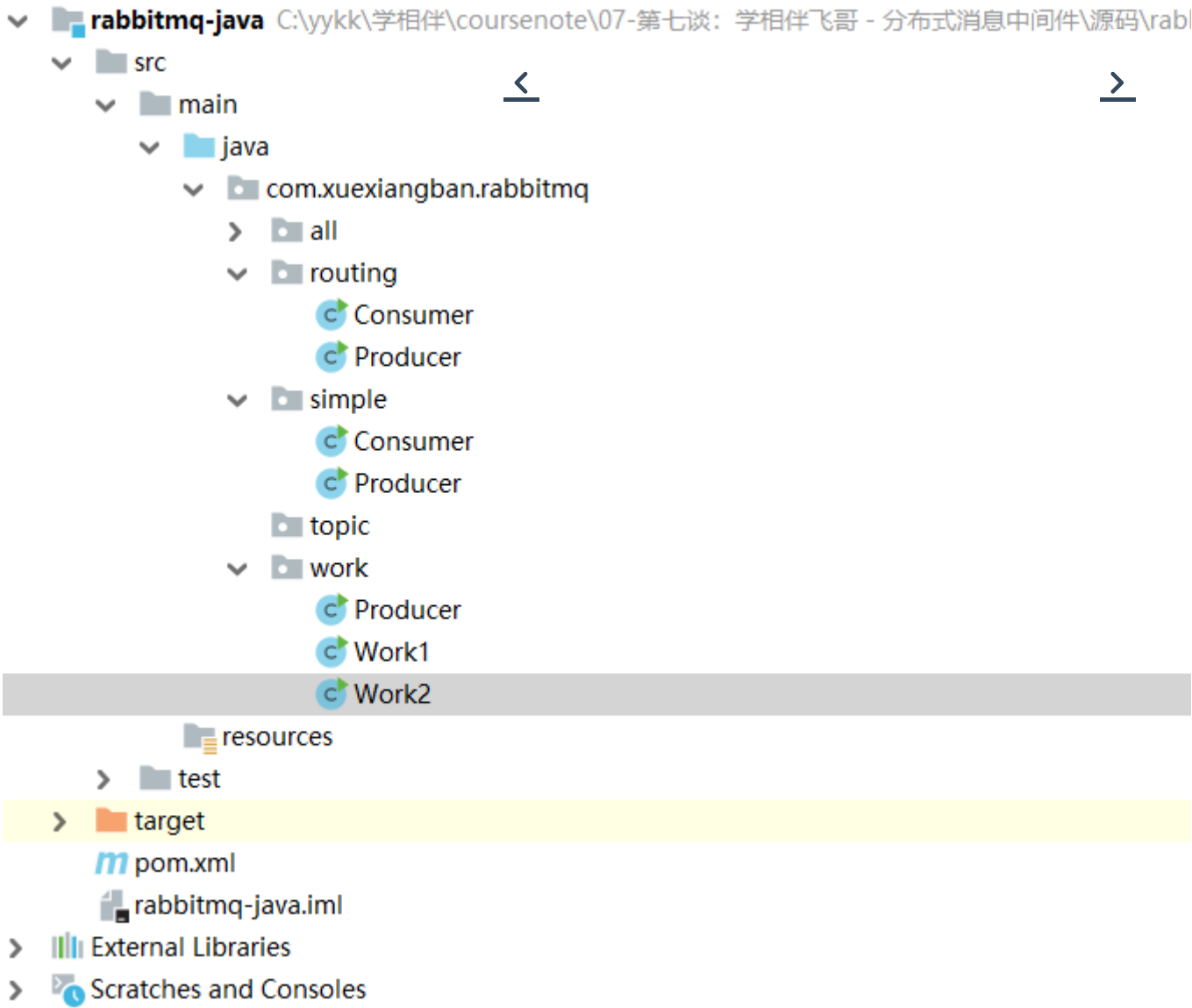
# RabbitMQ入门案例 - Simple 简单模式

飞哥 VIP 分类: 学习笔记 创建时间: 2021/03/02 14:53 字体 皮肤

## 01、实现步骤

- 1: jdk1.8
- 2: 构建一个maven工程
- 3: 导入rabbitmq的maven依赖
- 4: 启动rabbitmq-server服务
- 5: 定义生产者
- 6: 定义消费者
- 7: 观察消息的在rabbitmq-server服务中的过程

## 02、构建一个maven工程



## 03、导入rabbitmq的maven依赖

### 03-1: Java原生依赖

- 01、实现步骤
- 02、构建一个maven工程
- 03、导入rabbitmq的maven依赖
  - 03-1: Java原生依赖
  - 03-2: spring依赖
- 04、启动rabbitmq-server服务
- 05、定义生产者
- 06、定义消费者
- 07、观察消息的在rabbitmq-server服务中...

```
1. <dependency>
2.   <groupId>com.rabbitmq</groupId>
3.   <artifactId>amqp-client</artifactId>
4.   <version>5.10.0</version>
5. </dependency>
```



- [01、实现步骤](#)
- [02、构建一个maven工程](#)
- [03、导入rabbitmq的maven依赖](#) ▼
  - [03-1: Java原生依赖](#)
  - [03-2: spring依赖](#)
  - [04、启动rabbitmq-server服务](#)
  - [05、定义生产者](#)
  - [06、定义消费者](#)
  - [07、观察消息的在rabbitmq-server服务中...](#)

03-2: spring依赖

```
1. <dependency>
2.   <groupId>org.springframework.amqp</groupId>
3.   <artifactId>spring-amqp</artifactId>
4.   <version>2.2.5.RELEASE</version>
5. </dependency>
6.
7. <dependency>
8.   <groupId>org.springframework.amqp</groupId>
9.   <artifactId>spring-rabbit</artifactId>
10.  <version>2.2.5.RELEASE</version>
11. </dependency>
```

03-3、springboot依赖

```
1. <dependency>
2.   <groupId>org.springframework.boot</groupId>
3.   <artifactId>spring-boot-starter-amqp</artifactId>
4. </dependency>
```



上面根据自己的项目环境进行选择即可。



番外：rabbitmq和spring同属一个公司开放的产品，所以他们的支持也是非常完善，这也是为什么推荐使用ra

04、启动rabbitmq-server服务

```
1. systemctl start rabbitmq-server
2. 或者
3. docker start myrabbit
```

05、定义生产者





```
1. package com.xuexiangban.rabbitmq.simple;
2.
3. import com.rabbitmq.client.Channel;
4. import com.rabbitmq.client.Connection;
5. import com.rabbitmq.client.ConnectionFactory;
6.
7. /**
8.  * @author: 学相伴-飞哥
9.  * @description: Producer 简单队列生产者
10.  * @Date : 2021/3/2
11.  */
12. public class Producer {
13.
14.     public static void main(String[] args) {
15.
16.         // 1: 创建连接工厂
17.         ConnectionFactory connectionFactory = new ConnectionFactory();
18.         // 2: 设置连接属性
19.         connectionFactory.setHost("47.104.141.27");
20.         connectionFactory.setPort(5672);
21.         connectionFactory.setVirtualHost("/");
22.         connectionFactory.setUsername("admin");
23.         connectionFactory.setPassword("admin");
24.
25.         Connection connection = null;
26.         Channel channel = null;
27.         try {
28.             // 3: 从连接工厂中获取连接
29.             connection = connectionFactory.newConnection("生产者");
30.             // 4: 从连接中获取通道
31.             channel = connection.createChannel();
32.             // 5: 申明队列queue存储消息
33.             /*
34.              * 如果队列不存在，则会创建
35.              * Rabbitmq不允许创建两个相同的队列名称，否则会报错。
36.              *
37.              * @params1: queue 队列的名称
38.              * @params2: durable 队列是否持久化
39.              * @params3: exclusive 是否排他，即是否私有的，如果为true,会对当前队列加
40.              * 锁，并且连接自动关闭
41.              * @params4: autoDelete 是否自动删除，当最后一个消费者断开连接之后是否自动删除
42.              * @params5: arguments 可以设置队列附加参数，设置队列的有效期，消息的最大
43.              * 有效期等等。
44.              */
45.             channel.queueDeclare("queue1", false, false, false, null);
46.
47.             // 6: 准备发送消息的内容
48.             String message = "你好，学相伴！！！";
49.
50.             // 7: 发送消息给中间件rabbitmq-server
51.             // @params1: 交换机exchange
52.             // @params2: 队列名称/routing
53.             // @params3: 属性配置
54.             // @params4: 发送消息的内容
55.             channel.basicPublish("", "queue1", null, message.getBytes());
56.             System.out.println("消息发送成功!");
57.         } catch (Exception ex) {
58.             ex.printStackTrace();
59.             System.out.println("发送消息出现异常...");
60.         } finally {
61.             // 7: 释放连接关闭通道
62.             if (channel != null && channel.isOpen()) {
63.                 try {
64.                     channel.close();
65.                 } catch (Exception ex) {
66.                     ex.printStackTrace();
67.                 }
68.             }
69.         }
70.     }
71. }
```

01、实现步骤

02、构建一个maven工程

03、导入rabbitmq的maven依赖

03-1: Java原生依赖

03-2: spring依赖


04、启动rabbitmq-server服务

05、定义生产者

06、定义消费者

07、观察消息的在rabbitmq-server服务中...



- 01、实现步骤
- 02、构建一个maven工程
- 03、导入rabbitmq的maven依赖 
- 03-1: Java原生依赖
- 03-2: spring依赖
- 04、启动rabbitmq-server服务
- 05、定义生产者
- 06、定义消费者
- 07、观察消息的在rabbitmq-server服务中...

## 07、观察消息的在rabbitmq-server服务中...

**这里也可以查看到**



TOP

OverviewConnectionsChannelsExchangesQueuesAdmin

Queue queue1

Overview概览

Consumers消费者

Bindings绑定交换机，默认情况下：不指定会默认绑定一个默认的交换机exchange

Publish message发送消息

Get messages获取消息

Move messages移动消息

Delete删除消息

Purge

Runtime Metrics (Advanced)运行状况和统计

HTTP APIServer DocsTutorialsCommunity SupportCommunity SlackCommercial SupportP

- 01、实现步骤
- 02、构建一个maven工程
- 03、导入rabbitmq的maven依赖

03-1：Java原生依赖

03-2：spring依赖
- 04、启动rabbitmq-server服务
- 05、定义生产者
- 06、定义消费者
- 07、观察消息的在rabbitmq-server服务中...

3:进行预览和获取消息进行测试

Get messages

Warning: getting messages from a queue is destructive

Ack Mode: Nack message requeue true

Encoding: Nack message requeue true

Messages: Ack message requeue false

Reject requeue true

Reject requeue false

Get Message(s)

ack是应答模式:

1: nack 代表消息被消费不告

2: ack 应答, rabbitt-server接受到消费信息, 会从服务器短移除消息

06、定义消费者



```
1. package com.xuexiangban.rabbitmq.simple;
2.
3. import com.rabbitmq.client.Channel;
4. import com.rabbitmq.client.Connection;
5. import com.rabbitmq.client.ConnectionFactory;
6.
7. /**
8.  * @author: 学相伴-飞哥
9.  * @description: Producer 简单队列生产者
10.  * @Date : 2021/3/2
11.  */
12. public class Producer {
13.
14.     public static void main(String[] args) {
15.
16.         // 1: 创建连接工厂
17.         ConnectionFactory connectionFactory = new ConnectionFactory();
18.         // 2: 设置连接属性
19.         connectionFactory.setHost("47.104.141.27");
20.         connectionFactory.setPort(5672);
21.         connectionFactory.setVirtualHost("/");
22.         connectionFactory.setUsername("admin");
23.         connectionFactory.setPassword("admin");
24.
25.         Connection connection = null;
26.         Channel channel = null;
27.         try {
28.             // 3: 从连接工厂中获取连接
29.             connection = connectionFactory.newConnection("生产者");
30.             // 4: 从连接中获取通道
31.             channel = connection.createChannel();
32.             // 5: 申明队列queue存储消息
33.             /*
34.              * 如果队列不存在，则会创建
35.              * Rabbitmq不允许创建两个相同的队列名称，否则会报错。
36.              *
37.              * @params1: queue 队列的名称
38.              * @params2: durable 队列是否持久化
39.              * @params3: exclusive 是否排他，即是否私有的，如果为true,会对当前队列加
              问，并且连接自动关闭
40.              * @params4: autoDelete 是否自动删除，当最后一个消费者断开连接之后是否自
41.              * @params5: arguments 可以设置队列附加参数，设置队列的有效期，消息的最大
              期等等。
42.              */
43.             channel.queueDeclare("queue1", false, false, false, null);
44.
45.             // 6: 准备发送消息的内容
46.             String message = "你好，学相伴！！！";
47.
48.             // 7: 发送消息给中间件rabbitmq-server
49.             // @params1: 交换机exchange
50.             // @params2: 队列名称/routing
51.             // @params3: 属性配置
52.             // @params4: 发送消息的内容
53.             channel.basicPublish("", "queue1", null, message.getBytes());
54.             System.out.println("消息发送成功!");
55.         } catch (Exception ex) {
56.             ex.printStackTrace();
57.             System.out.println("发送消息出现异常...");
58.         } finally {
59.             // 7: 释放连接关闭通道
60.             if (channel != null && channel.isOpen()) {
61.                 try {
62.                     channel.close();
63.                 } catch (Exception ex) {
64.                     ex.printStackTrace();
65.                 }
66.             }
```

01、实现步骤

02、构建一个maven工程

03、导入rabbitmq的maven依赖

03-1: Java原生依赖

03-2: spring依赖

04、启动rabbitmq-server服务

05、定义生产者

06、定义消费者

07、观察消息的在rabbitmq-server服务中...



```
67.         if (connection != null) {
68.             try {
69.                 connection.close();
70.             } catch (Exception ex) {
71.                 ex.printStackTrace();
72.             }
73.         }
74.     }
75. }
76. }
```

## 07、观察消息的在rabbitmq-server服务中的过程

- 具体详细视频教程

[01、实现步骤](#)

[02、构建一个maven工程](#)

[03、导入rabbitmq的maven依赖](#)

>

[03-1：Java原生依赖](#)

[03-2：spring依赖](#)

[04、启动rabbitmq-server服务](#)

[05、定义生产者](#)

[06、定义消费者](#)

[07、观察消息的在rabbitmq-server服务中...](#)

[关于我们](#) | [加入我们](#) | [联系我们](#) | [帮助中心](#)

Copyright © 广东学相伴网络科技有限公司 [粤ICP备 - 2020109190号](#)

