生产者端

|  |
| --- |
| public class Producer {  private final static String QUEUE\_NAME = "reject\_queue";  public static void main(String[] args) throws IOException, TimeoutException {  // 获取链接  Connection connection = ConnectionUtil.getConnection();  // 创建信道  Channel channel = connection.createChannel();  // 声明队列，参数：队列名称，是否持久化，是否排他，是否自动删除，其它参数  channel.queueDeclare(QUEUE\_NAME, false, false, false, null);  String message = "Hello World";  // 发送消息 参数：交换机名称 这里使用的时默认交换机，队列名称或者是路由key，一些参数，消息体  channel.basicPublish("", QUEUE\_NAME, null, message.getBytes());  //关闭通道和连接(资源关闭最好用try-catch-finally语句处理)  channel.close();  connection.close();  }  } |

消费者端

|  |
| --- |
| public class Consumer {  private final static String QUEUE\_NAME = "reject\_queue";  public static void main(String[] args) throws IOException, TimeoutException {  // 获取链接  Connection connection = ConnectionUtil.getConnection();  // 创建信道  final Channel channel = connection.createChannel();  // 创建一个名称为ordinary\_queue，非持久化，非排他，自动删除的队列  channel.queueDeclare(QUEUE\_NAME, false, false, false, null);  // 通过匿名内部类实现消费方法  DefaultConsumer consumer = new DefaultConsumer(channel){  // 获取消息，并且处理，这个方法类似事件监听，如果有消息的时候，会被自动调用  /\*\*  \* 当接收到消息后此方法将被调用  \* @param consumerTag 消费者标签，用来标识消费者的，在监听队列时设置channel.basicConsume  \* @param envelope 信封，通过envelope  \* @param properties 消息属性  \* @param body 消息内容  \* @throws IOException  \*/  @Override  public void handleDelivery(String consumerTag, Envelope envelope, AMQP.BasicProperties properties, byte[] body) throws IOException {  // 第一个参数表示消息的序号，第二个参数表示消息是否重新入队  System.out.println("接收到消息，直接拒绝");  channel.basicReject(envelope.getDeliveryTag(), true);  //消息id，mq在channel中用来标识消息的id，可用于确认消息已接收  long deliveryTag = envelope.getDeliveryTag();  // body 即消息体  String msg = new String(body,"utf-8");  System.out.println(" [x] received : " + msg + "!");  }  };  // 监听队列，第二个参数：是否自动进行消息确认。  //参数：String queue, boolean autoAck, Consumer callback  /\*\*  \* 需要注意，自动ack是不会等消息处理完成，在自动应答模式下，  \* RabbitMQ一旦把消息传输给消费者后，服务器就默认为消息已经传送成功，并从队列中自动删除该消息  \* 参数明细：  \* 1、queue 队列名称  \* 2、autoAck 自动回复，当消费者接收到消息后要告诉mq消息已接收，如果将此参数设置为tru表示会自动回复mq，如果设置为false要通过编程实现回复  \* 3、callback，消费方法，当消费者接收到消息要执行的方法  \*/  channel.basicConsume(QUEUE\_NAME, true, consumer);  }  } |

Bug复现条件：只要消费者存在同时拒绝消息和确认消息，无论确认消息时手动还是自动，然后启动消费者，然后启动生产者，然后会发现队列没了。