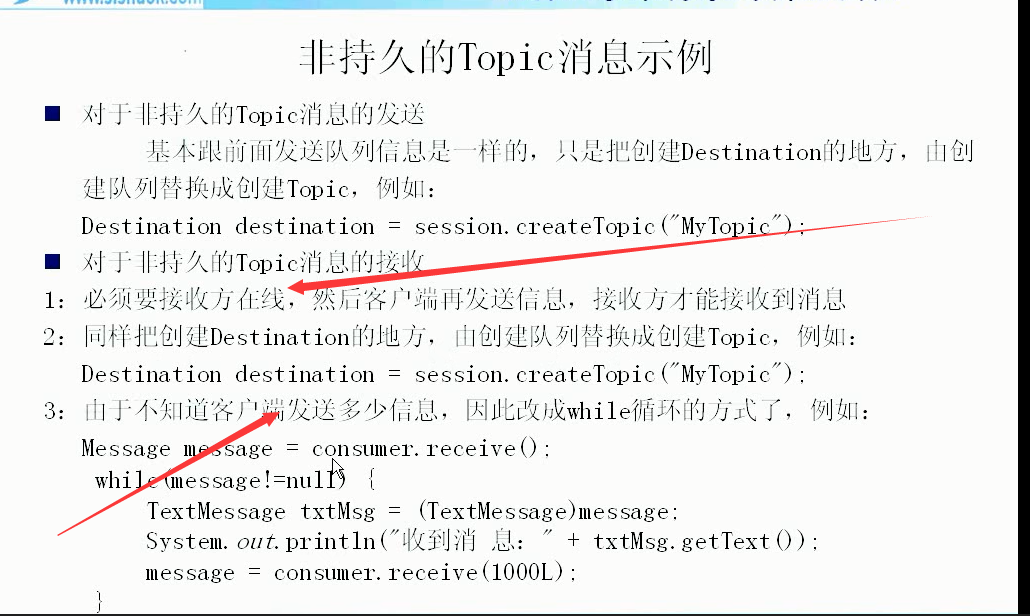
# 1、非持久的Topic消息

## 解释：

### 1、必须接收方在线，这个不会帮我们保存队列

### 2、由于不知道客户端发多少消息，因此改成while循环的方式读取



## 2、创建topic消息生产者

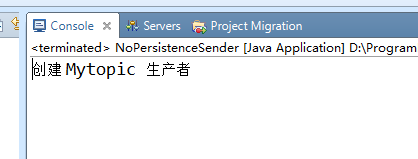
|  |
| --- |
| **public** **class** NoPersistenceSender {  **public** **static** **void** main(String[] args) **throws** Exception {  ConnectionFactory connectionFactory = **new** ActiveMQConnectionFactory(  "tcp://localhost:61616");    Connection connection = connectionFactory.createConnection();  connection.start();  Session session = connection.createSession(Boolean.*TRUE*,  Session.*AUTO\_ACKNOWLEDGE*);    Destination destination = session.createTopic("MyTopic");  System.*out*.println("创建 Mytopic 生产者");  MessageProducer producer = session.createProducer(destination);    **for** (**int** i = 0; i < 3; i++) {  TextMessage message = session.createTextMessage("message111--" + i);    // 通过消息生产者发出消息  producer.send(message);  }    session.commit();  session.close();  connection.close();  }  } |

## 3、创建消费者

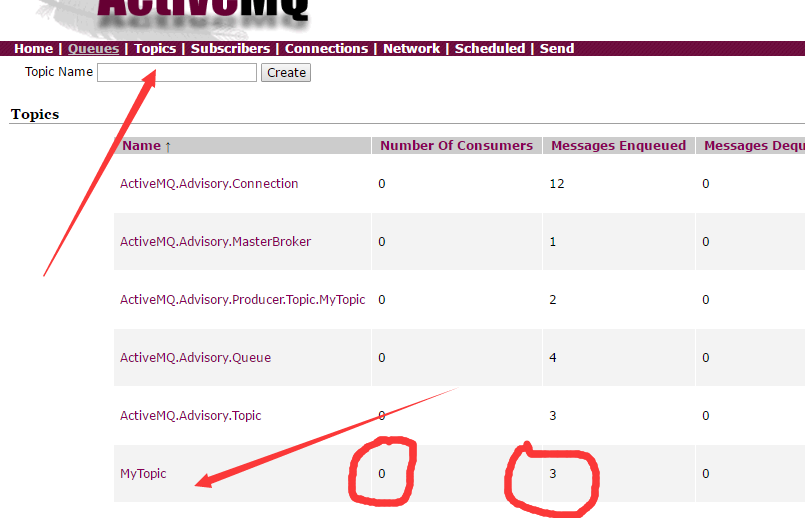
|  |
| --- |
| **public** **class** NoPersistenceReceiver {  **public** **static** **void** main(String[] args) **throws** Exception {  ConnectionFactory cf = **new** ActiveMQConnectionFactory(  "tcp://localhost:61616");    Connection connection = cf.createConnection();  connection.start();    **final** Session session = connection.createSession(Boolean.*TRUE*,  Session.*AUTO\_ACKNOWLEDGE*);    Destination destination = session.createTopic("MyTopic");      MessageConsumer consumer = session.createConsumer(destination);  System.*out*.println("创建 Mytopic 消费者");  Message message = consumer.receive();  **while**(message!=**null**) {  TextMessage txtMsg = (TextMessage)message;  System.*out*.println("收到消 息：" + txtMsg.getText());  message = consumer.receive(1000L);  }  session.commit();  session.close();  connection.close();  } |

## 4、接收方不再线（消费者没有运行），

### 1、运行生产者，控制台情况



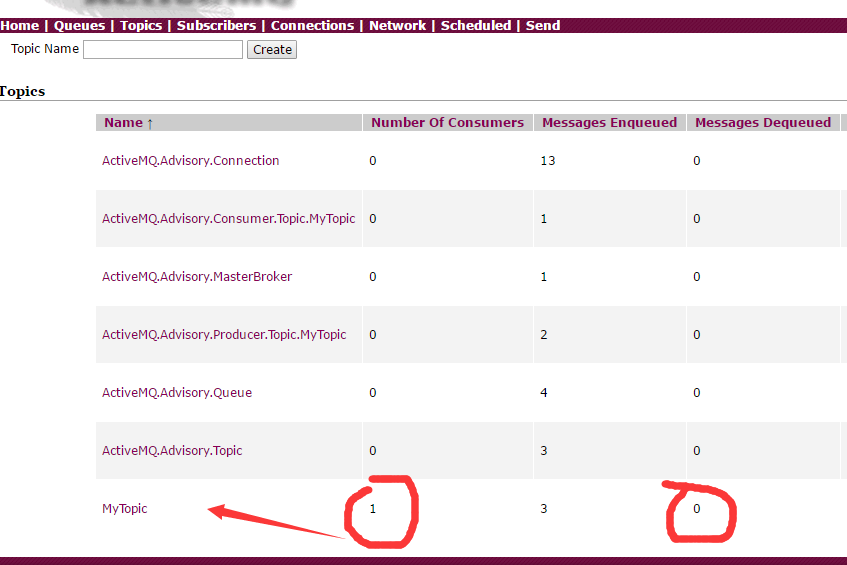
### 2、观察浏览器，会发现没有消费者，但是却有3条消息已经入队了



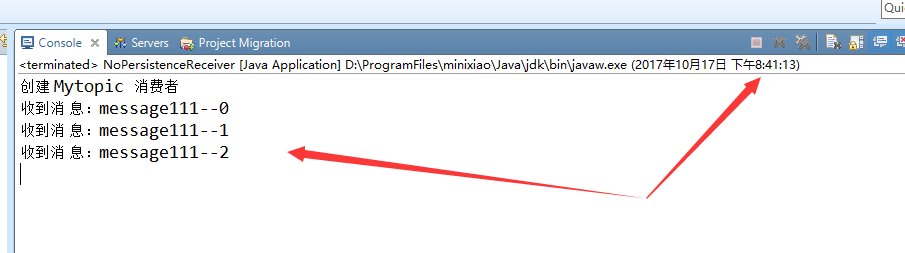
## 5、运行接收方-消费者，

### 1、观察控制台情况，控制台一直再等待，但是没有消息能够读取

### 2、观察浏览器，会看到有了消费者，但是却没有出队列



## 6、接收方在线（运行了消费者），再运行生产者，观察接收方控制台，会发现成功收到了消息,并且结束了控制台窗口





# 2、持久的topic消息（不会出队列，因为不止一个消费者会读取消息）





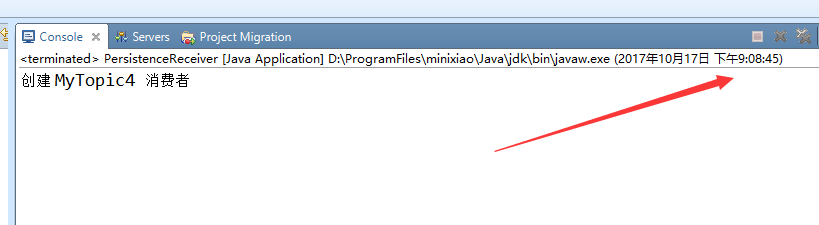
## 1、创建一个消费者

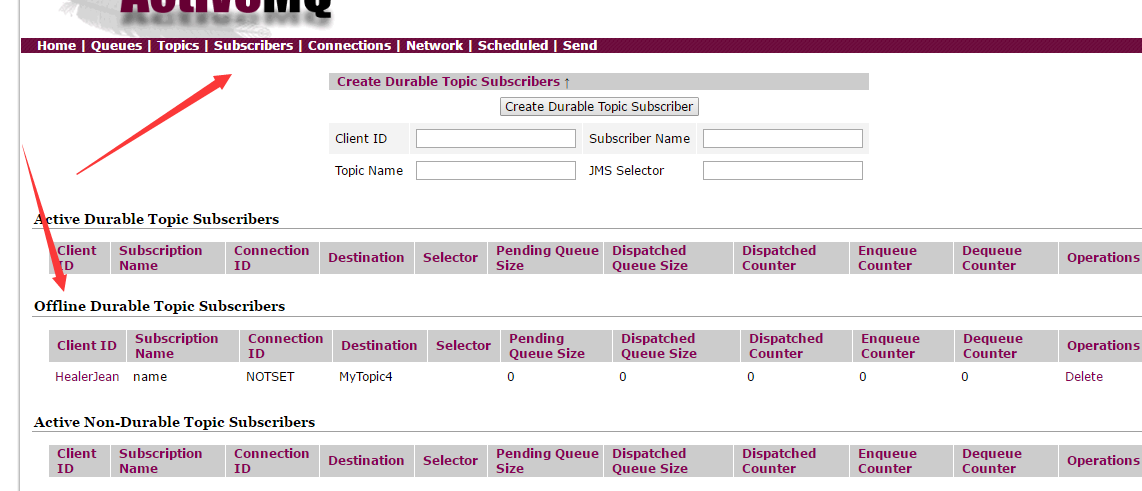
|  |
| --- |
| **public** **class** PersistenceSender {  **public** **static** **void** main(String[] args) **throws** Exception {  ActiveMQConnectionFactory connectionFactory = **new** ActiveMQConnectionFactory(  "tcp://localhost:61616");  Connection connection = connectionFactory.createConnection();    Session session = connection.createSession(Boolean.*TRUE*,  Session.*AUTO\_ACKNOWLEDGE*);    Destination destination = session.createTopic("MyTopic4");  System.*out*.println("创建 MyTopic4 生产者");  MessageProducer producer = session.createProducer(destination);  producer.setDeliveryMode(DeliveryMode.*PERSISTENT*);    connection.start();    **for** (**int** i = 0; i < 3; i++) {  TextMessage message = session.createTextMessage("message333--" + i);    // 通过消息生产者发出消息  producer.send(message);  }    session.commit();  session.close();  connection.close();  }  } |

## 2、创建一个生产者

|  |
| --- |
| **public** **class** PersistenceReceiver {  **public** **static** **void** main(String[] args) **throws** Exception {  ConnectionFactory cf = **new** ActiveMQConnectionFactory(  "tcp://localhost:61616");      Connection connection = cf.createConnection();  //设置连接客户端 id  connection.setClientID("HealerJean");    **final** Session session = connection.createSession(Boolean.*TRUE*,  Session.*AUTO\_ACKNOWLEDGE*);    Topic topic = session.createTopic("MyTopic4");  TopicSubscriber consumer = session.createDurableSubscriber(topic, "name");  System.*out*.println("创建 MyTopic4 消费者");  connection.start();    Message message = consumer.receive();  **while**(message!=**null**) {  TextMessage txtMsg = (TextMessage)message;  System.*out*.println("收到消 息：" + txtMsg.getText());  message = consumer.receive(1000L);  }  session.commit();  session.close();  connection.close();  }  } |

## 3、运行消费者，然后断开控制台，观察浏览器，会观察到有一个不在线的消费者客户端





## 3、运行生产者（会发现有3条消息入队了，而且有一个消费者（这里即使消费者以及关闭了，但是还是存在一个））



## 4、这个时候再运行消费者，就会读取消息了，但是不会出队的，因为别人也可以接收

