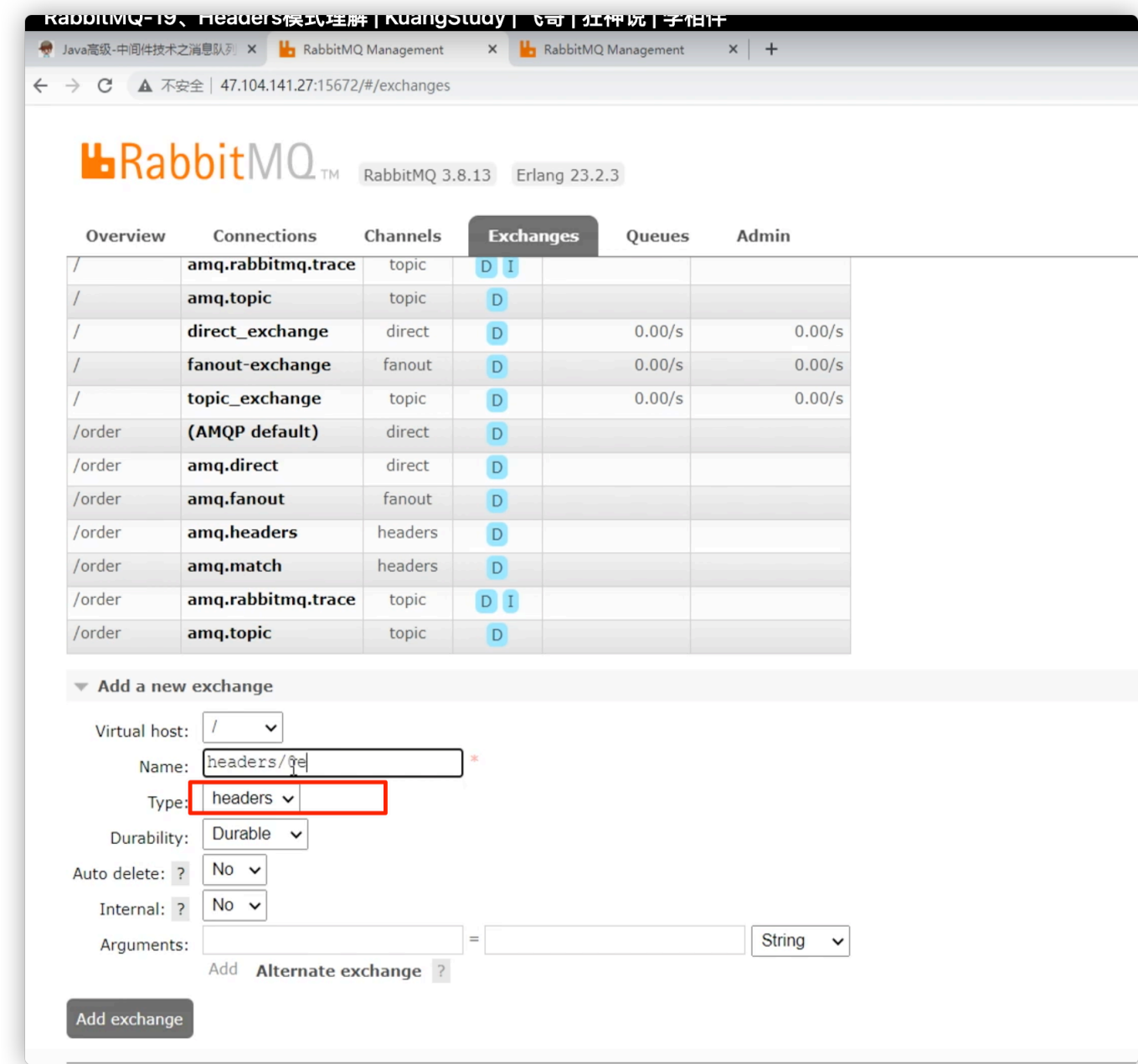


head模式 -- 不常用

界面模拟



Java高级-中间件技术之消息队列

RabbitMQ Management

RabbitMQ Management

+

← → ↻ ⚠ 不安全 | 47.104.141.27:15672/#/exchanges/%2F/headers_exchange

RabbitMQ™

RabbitMQ 3.8.13 Erlang 23.2.3

Overview

Connections

Channels

Exchanges

Queues

Admin

Add binding from this exchange

To queue ▾:

Routing key:

Arguments: = String ▾

Bind

▼ Publish message

Routing key:

Headers: ? x = 1 String ▾

String ▾

Properties: ?

Payload: hello queueul

Publish message

▼ Delete this exchange

工作模式总结

- 主要有两种模式：
- 1、轮询模式的分发：一个消费者一条，按均分配；

2、公平分发：根据消费者的消费能力进行公平分发，处理快的处理的多，处理慢的处理的少；按劳分配；

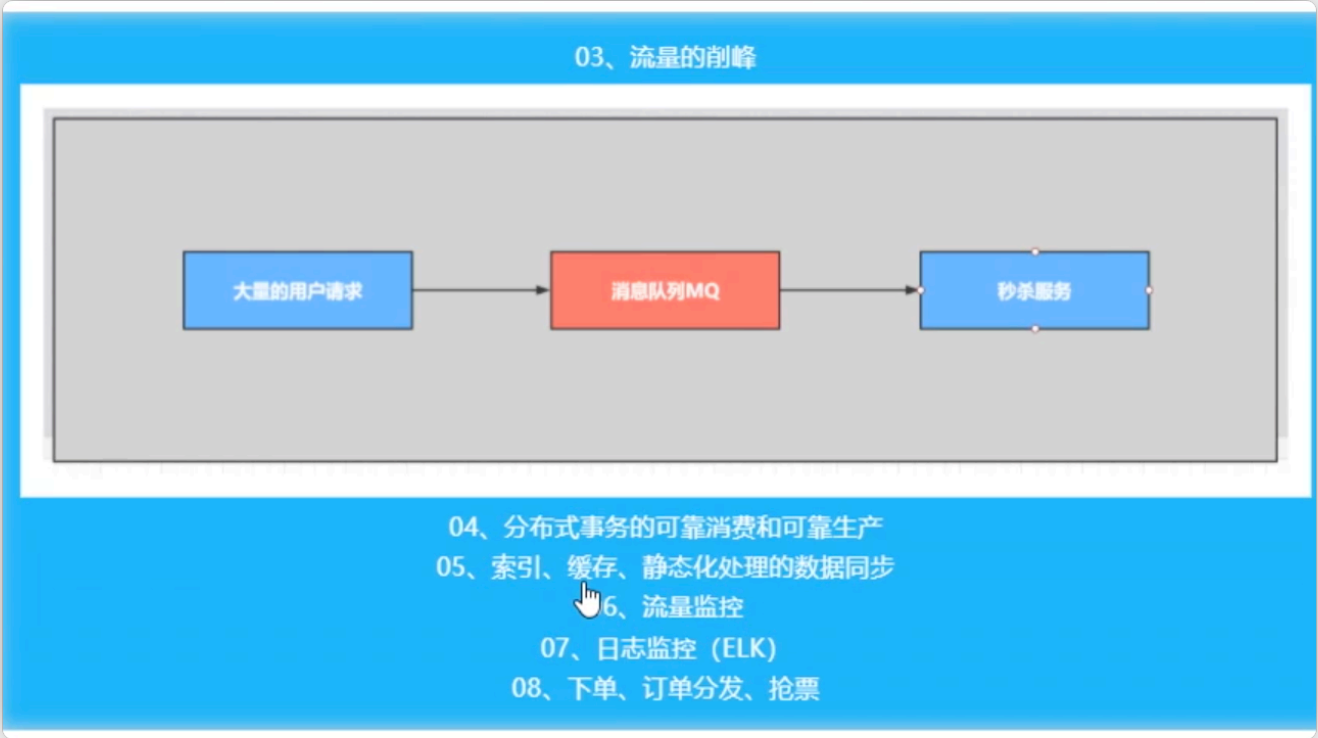
轮询使用的是自动应答

公平分发必须是手动应答



```
39  */
40
41  // 这里如果queue已经被创建过一次了, 可以不需要定义
42  //channel.queueDeclare("queue1", false, true, false, null);
43  // 同一时刻, 服务器只会推送一条消息给消费者
44  //channel.basicQos(1);
45  // 6: 定义接受消息的回调
46  Channel finalChannel = channel;
47  //finalChannel.basicQos(1);
48  finalChannel.basicConsume( queue: "queue1", autoAck: false, new DeliverCallback() {
49      @Override
50      public void handle(String s, Delivery delivery) throws IOException {
51          try{
52              System.out.println("Work2-收到消息是: " + new String(delivery.getBody()));
53              Thread.sleep( millis: 200);
54              finalChannel.basicAck( delivery.getEnvelope().getDeliveryTag(), multiple: false);
55          } catch (Exception ex) {
56              ex.printStackTrace();
57          }
58      }, new CancelCallback() {
59          @Override
60          public void handle(String s) throws IOException {
61              //
62          }
63      }
64  });
```

mq的使用场景



07

08

04、分布式事务的可靠消费和可靠生产

09

05、索引、缓存、静态化处理的数据同步

10

06、流量监控

I

11

07、日志监控 (ELK)

12

08、下单、订单分发、抢票

13

01、解耦、削峰、异步