消费者端如何接收有序消息

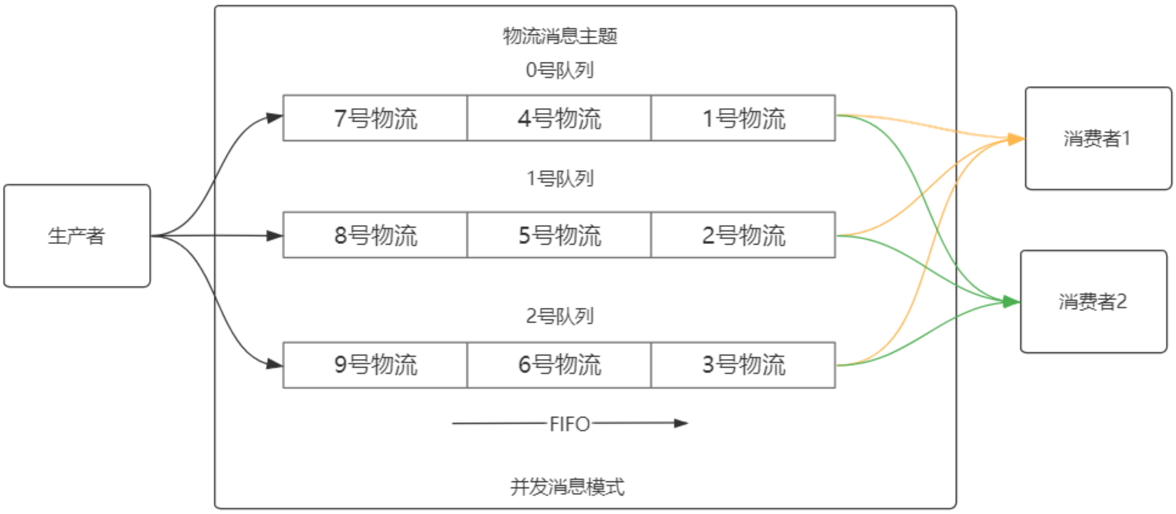
队列消费的两种模式

并发消费模式

当同一类消息被送入不同队列，且这些消息在处理上并不需要按时序消费时，可以考虑使用并发消费模式。

并发消费模式生产者会将消息轮询发送到不同的队列当中，这些队列会和消费者实例建立多个连接（线程）将消息并发送入到不同的消费者。因为消费者处理速度有快慢，所以并不能保证物流数据会按0~9的顺序依次消费。

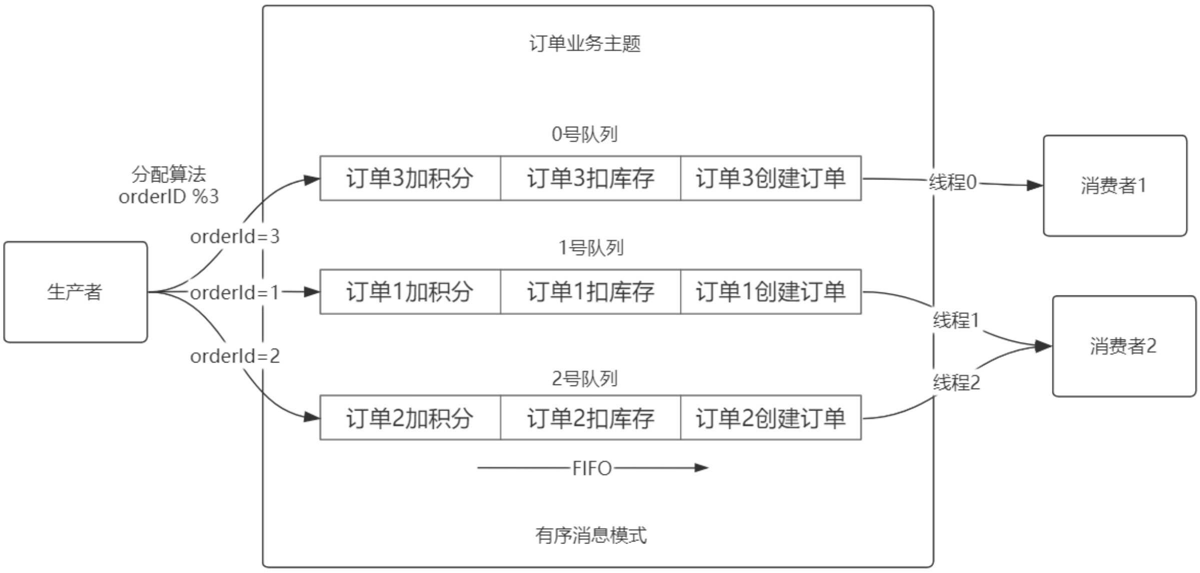
并发消费模式处理效率很高，但无法保证有序性。



有序消费模式

有序消息是指生产者在产生数据的时候，根据Hash规则指定让消息放入哪个队列，在消费者消费时会保证不同消费者针对每一个队列只有唯一的连接（线程）用于消费指定队列。

有序消费模式可以保证消息按队列FIFO顺序依次被消费，但因此失去并发性能，有序消费模式只有在业务要求必须按顺序消费的场景下才允许使用。



RocketMQ如何实现有序消息

要实现RocketMQ有序消息需要两点调整：

- 生产者端要求按id等唯一标识分配消息队列
- 消费者端采用专用的监听器保证对队列的单线程应用

下面咱们来看一下代码：

生产者端

com.itlaoqi.rocketmq.seqmsg.SequenceMessageProvider

核心代码是在向Broker发送消息时附加MessageQueueSelector对象，在实现select方法时指定存放到哪个队列中。

```
@Slf4j
//发送分区顺序消息
public class SequenceMessageProvider {
    public static void main(String[] args) {
        //前置准备代码
        DefaultMQProducer producer = new DefaultMQProducer("producer-group");
        producer.setNamesrvAddr("192.168.31.103:9876");
        try {
            producer.start();
            //模拟10笔订单
            for (Integer orderId = 1; orderId <= 10; orderId++) {
                //每笔订单要发送3条消息：(1)创建订单 (2)订单库存扣减 (3)加积分
                for(int i = 0 ; i < 3 ; i++) {
                    String data = "";
                    switch (i % 3){
                        case 0:
                            data = orderId + "号创建订单";
                            break;
                        case 1:
                            data = orderId + "号订单减库存";
                            break;
                        case 2:
                            data = orderId + "号订单加积分";
                            break;
                    }
                    //创建消息对象 topic="order",tags="order",key=orderId
                    Message message = new Message("order","order",orderId.toString(),
data.getBytes("UTF-8"));
                    //发送消息，实现MessageQueueSelector接口
                    SendResult result = producer.send(message, new
MessageQueueSelector() {
                        //select方法决定向broker哪一个队列发送消息
                        @Override
                        public MessageQueue select(List<MessageQueue> mqs, Message
msg, Object arg) {
                            int orderId = Integer.parseInt(msg.getKeys());
                            int index = orderId % mqs.size();
                            MessageQueue messageQueue = mqs.get(index);
                            log.info("id:{},data:{},queue:{}", orderId ,new String(msg.getBody()),
messageQueue);
                            return messageQueue;
                        }
                    }, null);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                producer.shutdown();
                System.out.println("连接已关闭");
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}
```

运行结果

```
id:1,data:1号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a,
queueId=1]
id:1,data:1号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a,
queueId=1]
id:1,data:1号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a,
queueId=1]
id:2,data:2号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a,
queueId=2]
```

```
id:2,data:2号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:2,data:2号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:3,data:3号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:3,data:3号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:3,data:3号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:4,data:4号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
id:4,data:4号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
id:4,data:4号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
id:5,data:5号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:5,data:5号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:5,data:5号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:6,data:6号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:6,data:6号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:6,data:6号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:7,data:7号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
id:7,data:7号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
id:7,data:7号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
id:8,data:8号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:8,data:8号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:8,data:8号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=2]
id:9,data:9号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:9,data:9号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:9,data:9号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=0]
id:10,data:10号创建订单,queue:MessageQueue [topic=order, brokerName=broker-a,
queueId=1]
id:10,data:10号订单减库存,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
id:10,data:10号订单加积分,queue:MessageQueue [topic=order, brokerName=broker-a, queueId=1]
```

消费者端

com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer

消费者端最大的变化是registerMessageListener监听器要实例化MessageListenerOrderly对象，用于为每一个队列分配唯一的连接（线程）进行消费。

每一批消息从Broker投递给消费者都会触发consumeMessage()方法实现对消息的消费。

```
@Slf4j
public class SequenceMessageConsumer {
    public static void main(String[] args) throws Exception {
        // 声明并初始化一个 consumer
        // 需要一个 consumer group 名字作为构造方法的参数
        DefaultMQPushConsumer consumer = new DefaultMQPushConsumer("consumer-group");
        // 同样也要设置 NameServer 地址，须要与提供者的地址列表保持一致
        consumer.setNamesrvAddr("192.168.31.103:9876");
        // 设置 consumer 所订阅的 Topic 和 Tag，*代表全部的 Tag
        consumer.subscribe("order", "*");
        // 注册消息监听者，消费者端要增加MessageListenerOrderly监听器，用于实现有序队列
        consumer.registerMessageListener(new MessageListenerOrderly() {
            @Override
            public ConsumeOrderlyStatus consumeMessage(List<MessageExt> list, ConsumeOrderlyContext context) {
                //遍历输出
                list.forEach(msg->{
                    log.info("{},{},{}",msg.getKeys(),new String(msg.getBody()),context.getMessageQueue());
                });
                //确认接收成功
                return ConsumeOrderlyStatus.SUCCESS;
            }
        });
        // 启动消费者
        consumer.start();
        log.info("消费者启动成功，正在监听新消息");
    }
}
```

这里我们启动两个实例，查看运行结果：

实例1输出日志，可以发现所有queueId=2的队列消息都被实例1的ConsumeMessageThread_1线程消费，采用FIFO依次进行处理，同一个队列中的消息消费是有序的。

```
15:58:28.611 [main] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 消费者启动成功，正在监
听新消息
15:59:49.457 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 2,2号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 2,2号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 2,2号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 5,5号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 5,5号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 5,5号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 8,8号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 8,8号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
15:59:49.460 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 8,8号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=2]
```

实例2输出日志，可以发现所有queueId=1的队列消息都被实例2的ConsumeMessageThread_1线程消费，采用FIFO依次进行处理；

所有queueId=0的队列消息都被实例2的ConsumeMessageThread_0线程消费，采用FIFO依次进行处理，同一个队列中的消息消费是有序的，不同队列间不能保证消息有序。

```
[main] INFO com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 消费者启动
成功，正在监听新消息
15:59:31.370 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 1,1号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.370 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 3,3号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 3,3号订单减库
```

```
存,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 1,1号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 3,3号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 1,1号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 6,6号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 4,4号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 6,6号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 4,4号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 6,6号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 4,4号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 9,9号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 7,7号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 9,9号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 7,7号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 9,9号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 7,7号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
```

```
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 3,3号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 10,10号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 3,3号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 10,10号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 3,3号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 10,10号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 6,6号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 1,1号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 6,6号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 1,1号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 6,6号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 1,1号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 9,9号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 4,4号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 9,9号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 4,4号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
```

```
15:59:31.373 [ConsumeMessageThread_2] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 9,9号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=0]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 4,4号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.373 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 7,7号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.374 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 7,7号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.374 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 7,7号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.374 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 10,10号创建订
单,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.374 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 10,10号订单减库
存,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
15:59:31.374 [ConsumeMessageThread_1] INFO
com.itlaoqi.rocketmq.seqmsg.SequenceMessageConsumer - 10,10号订单加积
分,MessageQueue [topic=order, brokerName=broker-a, queueId=1]
```

FAQ

如何实现消息全局顺序消费?

只需要在生产者固定将所有消息发送到0号队列即可保证全局有序，这也意味着全局采用单线程消费，执行效率极差。

```
@Override
public MessageQueue select(List<MessageQueue> mqs, Message msg, Object arg)
{
    MessageQueue messageQueue = mqs.get(0);
    return messageQueue;
}
```

有序消费有什么使用限制吗?

有序消费模式只支持集群模式（CLUSTERING），不支持广播模式（BROADCASTING），采用广播模式会无法接收到数据。

```
//设置为集群模式
consumer.setMessageModel(MessageModel.CLUSTERING);//支持有序消息，默认模式
consumer.setMessageModel(MessageModel.BROADCASTING);//不支持有序消息
```