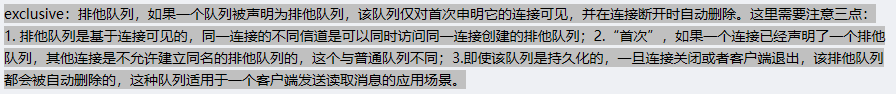
## 如何保证消息有序

如下场景，停车场收费系统与云端计费系统的通讯，车辆识别出场时发送消息的顺序为：待付款-》线下优惠券上传-》付款。优惠券上传必须在付款之前。

方案一（适合简单的客户端场景，无高并发，例如停车场收费端）：

* 消息发布者同步发送消息；
* 只有一个消费者；

使用到的技术，zookeeper选主消费者；exclusive consumer保证同一时刻只有一个消费者。

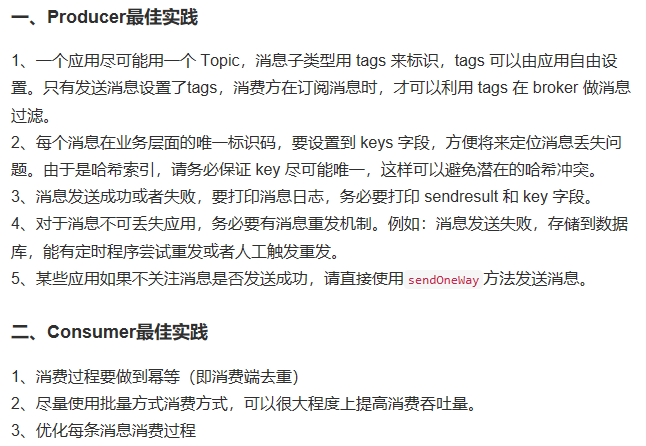


改进版，同一用户的消息发送到同一消息队列中，这样就可以使用多队列多消费者了。

方案二（高并发高可用）：

* 要保持多个消息之间的时间顺序，首先它们要有一个全局的时间顺序。因此，每个消息在被创建时，都将被赋予一个全局唯一的、单调递增的、连续的序列号（SerialNumber，SN）。可以通过一个全局计数器来实现这一点。通过比较两个消息的SN，确定其先后顺序。

## 消息队列的最佳实践



怎样异步和同步生成消息

批量消费和发送

镜像队列实现高可用

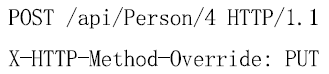
分布式事务

## 设计Restful api注意事项

1. Restful api的url是以动词+名词的形式存在，HTTP动词+资源名词（复数）

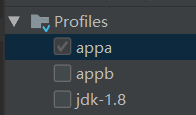


1. 转账行为是作为一种服务资源存在，其url是Get http://xxx/transfer
2. 有些客户端只能使用get和post两种方法，服务器必须接受post模拟其他三种方法，这是客户端发出的http请求要加上X-HTTP-Method-Override属性，告诉服务器使用哪个动词

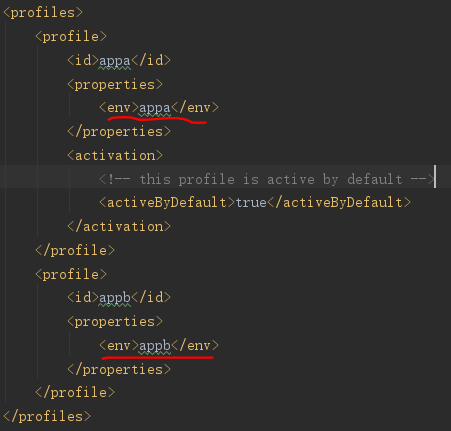


## Springboot的yml配置文件切换

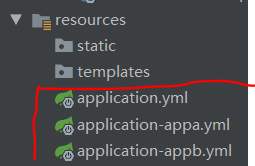
想要得到的效果是在IDE中点击需要切换的Profile就可以按这个profile来运行或打包程序



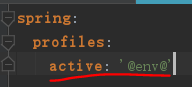
首先需要在pom里面设定本项目有多少不同的profile配置文件



其次在项目中添加具体的配置文件

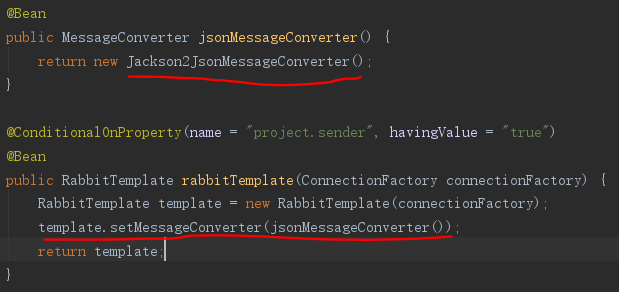


最好需要yml读取maven变量（关键）

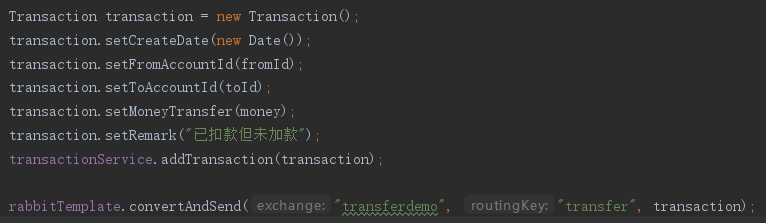


## 生产者和消费者之间传json对象

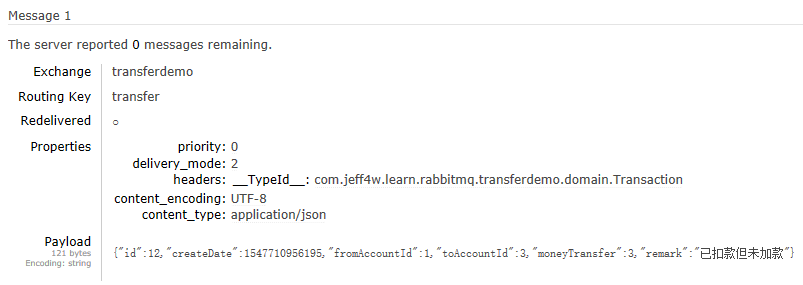
* Rabbitmq允许在生产者和消费者之间传java对象（序列化和反序列化）或json，它们底层都是传字节码。建议传json对象，如果传java对象需要使用MessageConverter来为发送和接收序列化和反序列化，MessageConverter接收的是Object，所以需要把具体的对象类名也要传给消费者，好让消费者根据这个类名反序列化。因为消费者和生产者一般是放在两个项目中的，所以其路径和类名都可能不一样，这样传送到消费者端可能会找不到类。
* 传送json时的配置，rabbitmq提供了json的默认消息转换器Jackson2JsonMessageConverter，并把它设置给rabbittemplate



* 应用可直接把对象发送出去，不需要自己转json



* Broker接收到json格式的消息

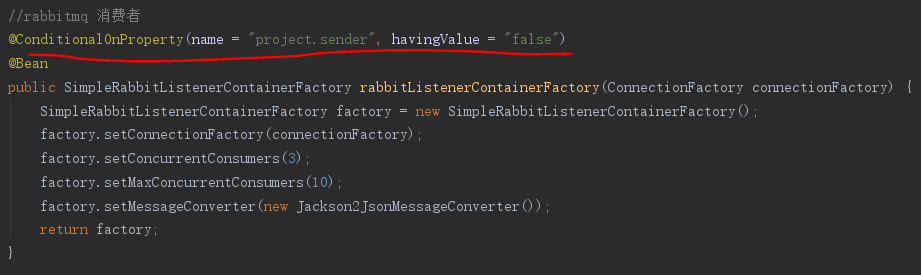


## 通过选择maven profile来选择配置哪种组件

本项目包含rabbitmq的消息发送方和接受方两个应用，且发送方的组件与接收方不一样且同一时刻只能存在一方。使用@ConditionalOnProperty根据配置文件来确定配置哪些组件。

当配置文件中的project.sender值等于false时，配置SimpleRabbitListenerContainerFactory组件

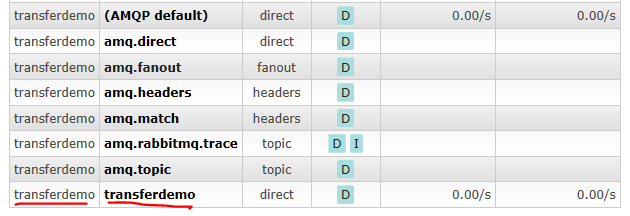




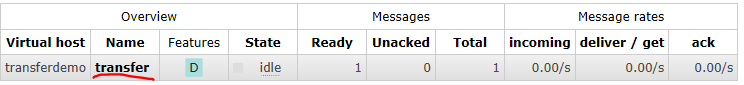
## 创建先于应用的持久化exchanger、queue

本项目场景需要提前设置好的交换机和队列，并且是持久化的，即即使消息发送方和接收方不存在的情况下照样存在。

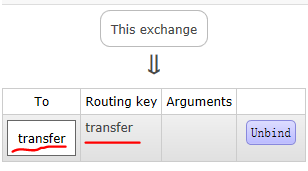
1. 使用rabbitmq的管理页面创建名叫transferdemo的虚拟机和transferdemo的交换机



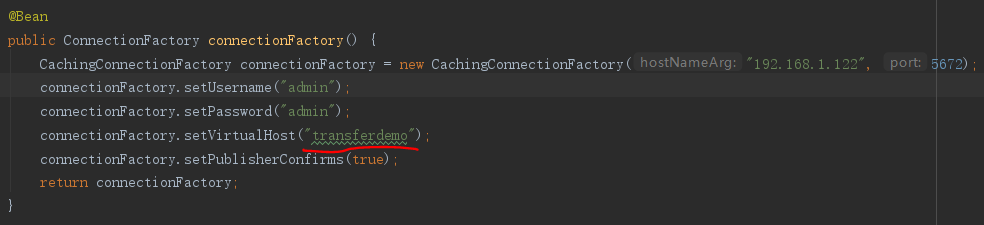
1. 创建队列transfer



1. 把队列绑定到交换机上



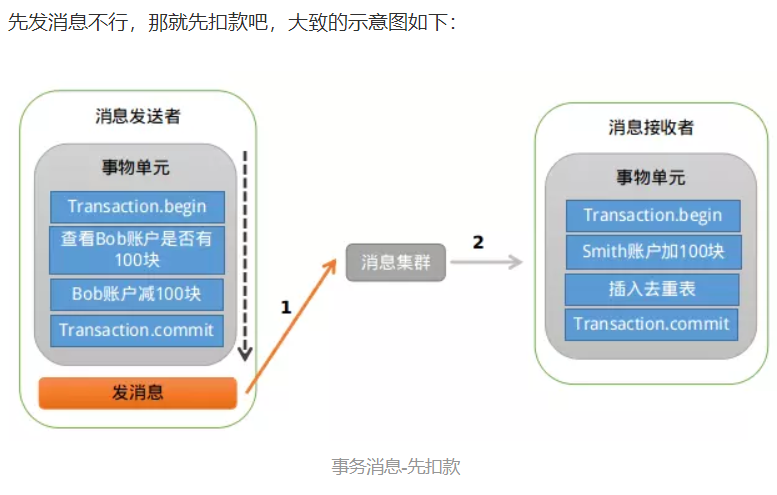
1. 发送方不需要再次声明交换机和队列，只需指定即可





## 消息队列事务场景

首先说一下数据库事务，在j2ee应用中，业务逻辑分布在service层且多为数据库操作，所以事务范围是包含整个service方法。数据库事务回滚的前提条件就是在事务控制范围内捕捉到RuntimeException。如果service内既有数据库业务又有消息队列业务，则需要把数据库业务提前消息业务在后，这样消息业务出错可以抛RuntimeException来回滚整个service方法。



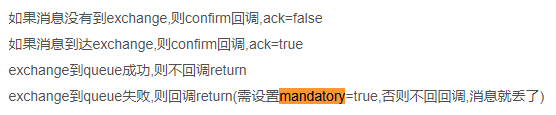
以上场景可以得出消息的事务并没有卵用。

## Rabbitmq的消息确认机制

分两类，一种是消息发送确认，一种是消费接收确认。

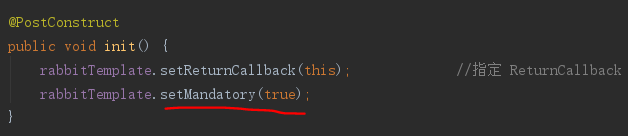
* 消息发送确认：

如果不使用rabbitmq的事务，就必须保证消息能够发送到broker端。Rabbitmq提供了confirmcallback和returncallback两个异步响应接口，作用分别如下：



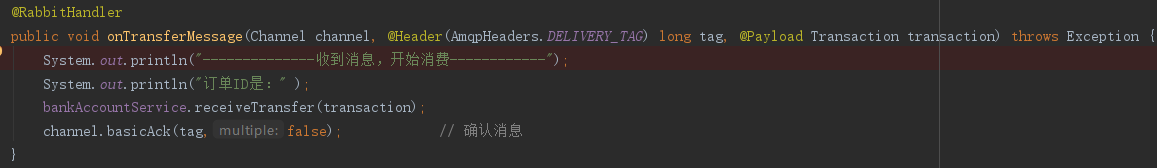
本项目使用returncallback，因为需要保证消息发送到指定的队列才算成功，设置如下：





* 消费接收确认





## Rabbitmq消息的推、拉模式

常用模式是推，但推不能批处理消息。

## 分布式应用，银行转账

由于账号巨量存在，账号存在在多个服务器上。如果分属两个不同服务器上的账户要求转账，则属于分布式事务行为。本场景需要实现的是，

* 首先扣减a应用上账户的钱，扣减后再发送消息给b应用增加其账号钱，需要保证扣减a账号钱和发送消息到b两个业务同时成功或同时失败（事务）。

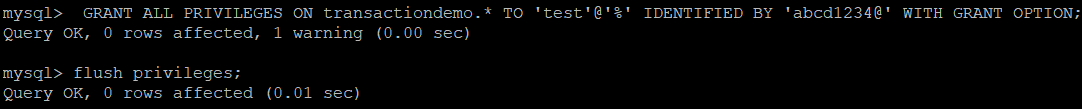
1. 扣减a账户的钱与把发送的消息transaction记录入数据库合一起作为数据库事务；
2. 异步发送消息transaction，使用confirmcallback接收broker的ack。如收到ack则删除数据库上的transaction消息，如收不到ack则重发数据库上的transaction消息。
3. 如果多次重发都失败，则人工干预。

* Broker发送消息transaction给b应用的消费者，消费者开事务增加b账号的钱，成功则发确认消息给broker。

1. 创建数据库transactiondemo数据库给a应用，transactiondemo2数据库给b应用







1、建立一个转账应用，但分别两个实例连接两个不同的数据库

## 问题：

Consumer不能自动删除？？？？

