# SpringAMQP实现工作消息队列Work Queue

上一节已经学习了如何基于SpringAMQP实现基本消息队列Basic Queue中的消息发送与消息接收！下面学习如何基于SpringAMQP实现工作消息队列Work Queue中的消息发送与消息接收!

## Work Queue结构

先看一下Work Queue的结构，可以看到同样具备消息的发送者、对列、消息的消费者，但是队列有多个消费者监听，这就是和之前基本队列的区别：即多个消费者监听同一个队列！

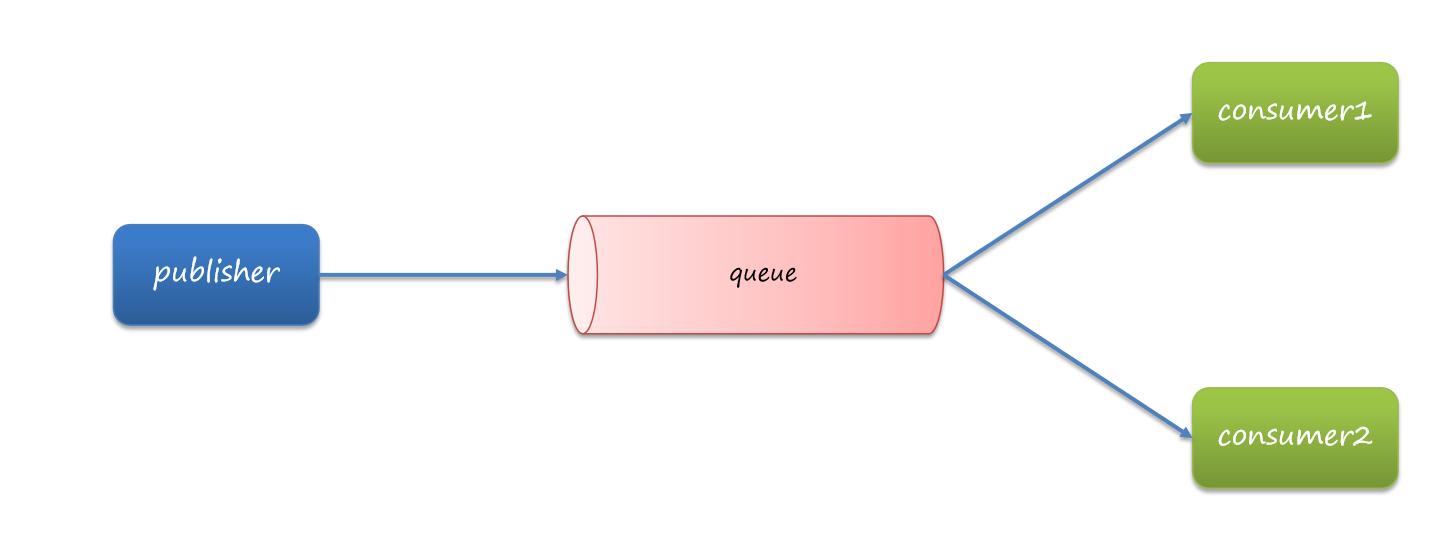
有多个消费者，那将来队列中的消息给谁呢？我们知道在RabbitMQ中，消息是阅后即焚的！即队列中的消息一旦给了消费者1，队列中的消息就会删除，消费者2不会拿到这个消息！所以如果队列中有50个消息，不可能每个消费者都接收消费50条，应该是每个消费者加起来接收消费50条！即在Work Queue工作队列中，消费者具有合作关系，共同处理消息！

为什么要多个消费者？

假设你真的只有一个consumer，而这个consumer的处理速度只有每秒40条，而Publisher每秒却发送50条，消费者处理不过来，那么每秒钟就会多出10条消息没人处理，那这些消息只能堆积在队列当中，队列在内存中有存储上限的，每秒多10条，假以时日，是会把队列堆满，这个时候再有消息就过不来，消息就会被丢弃掉！这就出问题了，而如果我有两个Consumer，那么就不会有这个问题！

这就是要挂两个消费者的原因了，也就是说这个Work Queue工作对列，并不是真的有一个新的队列类型，和基本队列Basic Queue没什么不同，只不过在设计的时候队列挂了多个消费者，而不是只有一个消费者。

这种情况一般时消费者处理消息能力不足的场景！提高消息处理的速度，避免消息的堆积！



## 实现基本思路

1：在publisher服务中定义测试方法，每秒产生50条消息，发送到simple.queue中

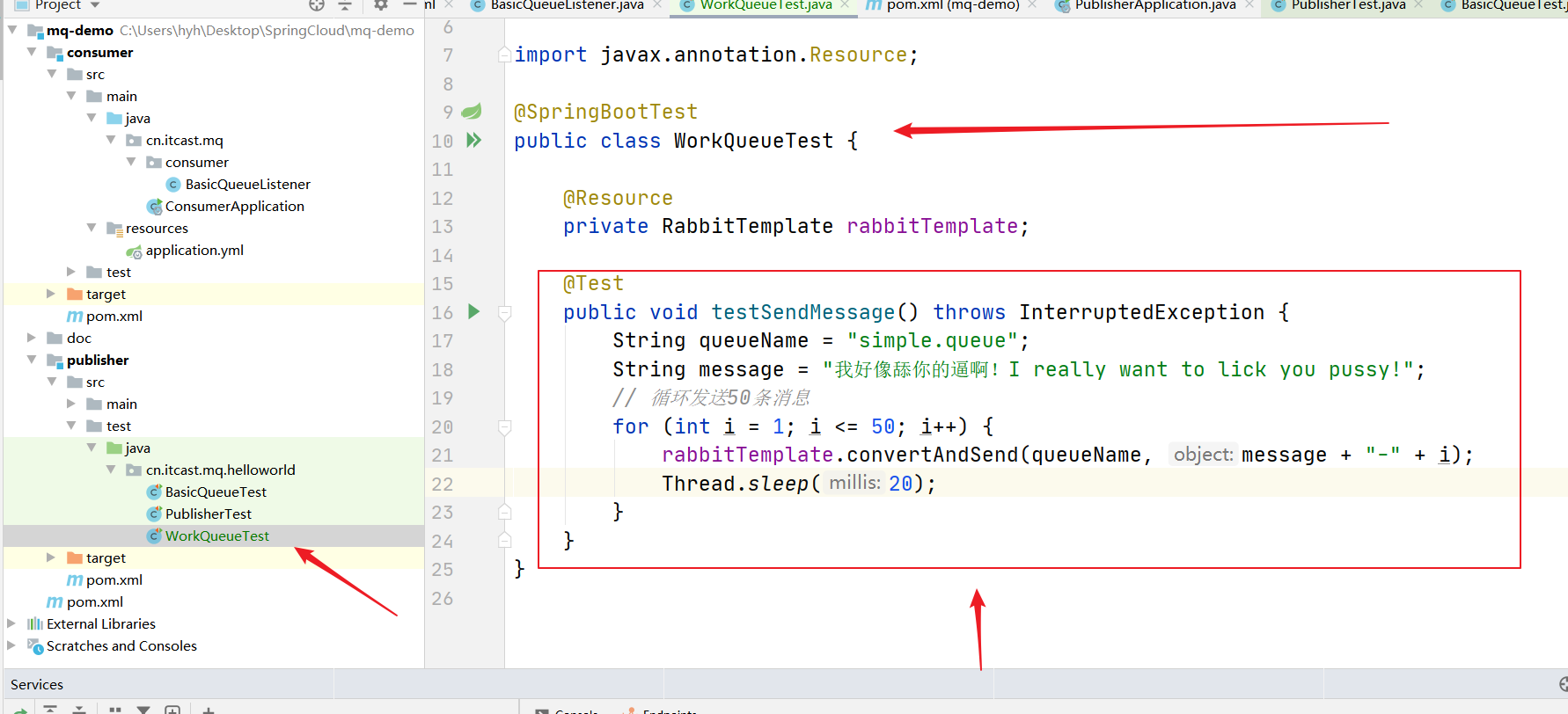
2：在consumer服务中定义两个消息监听者，都监听simple.queue对类

3：消费者1每秒处理50条消息，消费者2每秒处理10条消息，消费者1和消费者2每秒加起来能处理60条消息，超出了publisher每秒的发送频率！理论上两个消费者应该能在1秒内把publisher发送的消息处理完

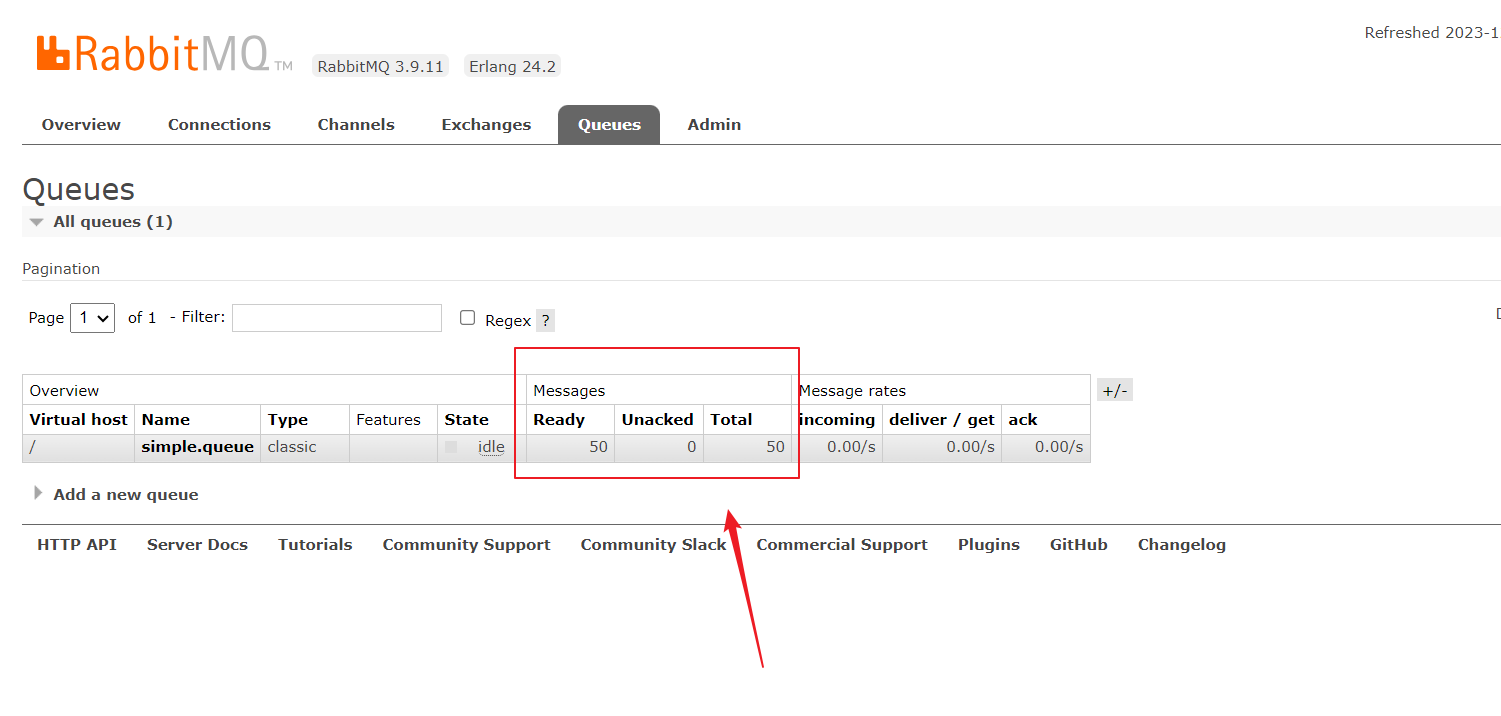
## 发送消息

### 定义发送消息的方法

和基本队列发送消息没什么不同！不过要演示一个消费者不能完全处理消息的场景，设计每秒发50条消息

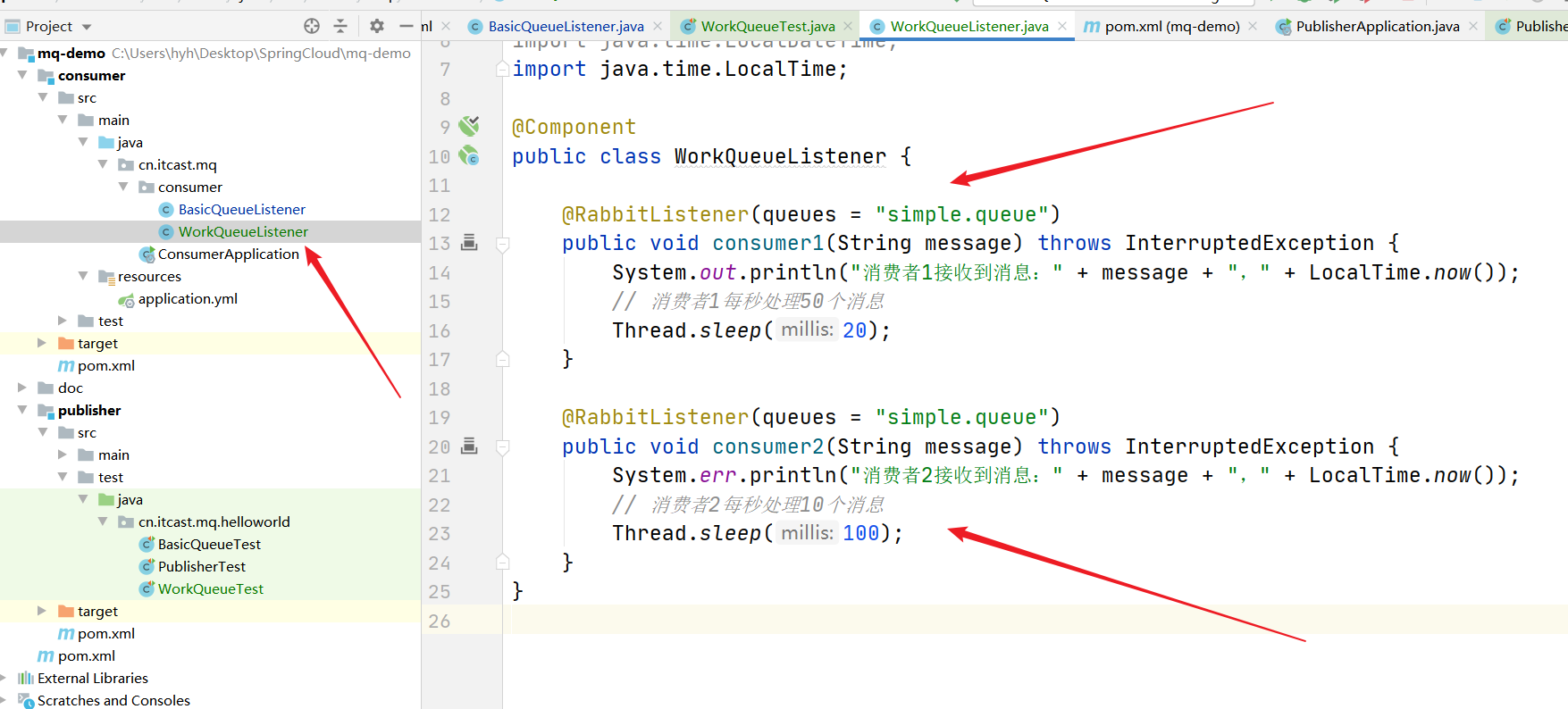


### RabbitMq管理后台查看消息

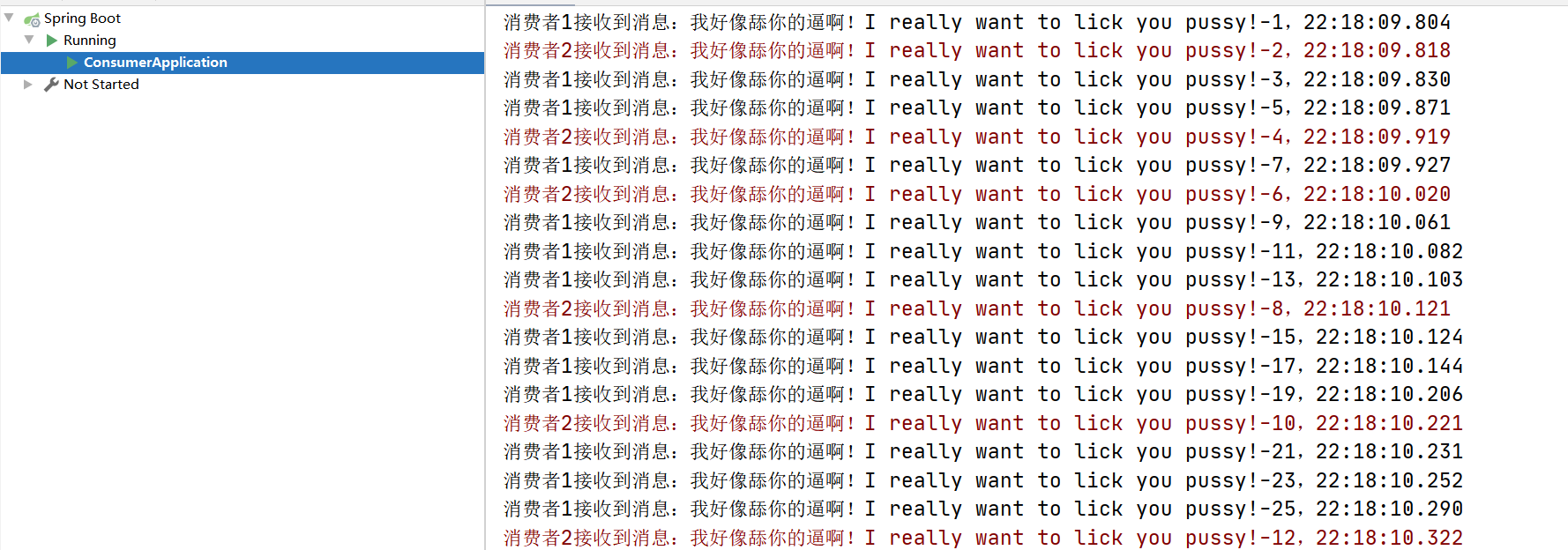


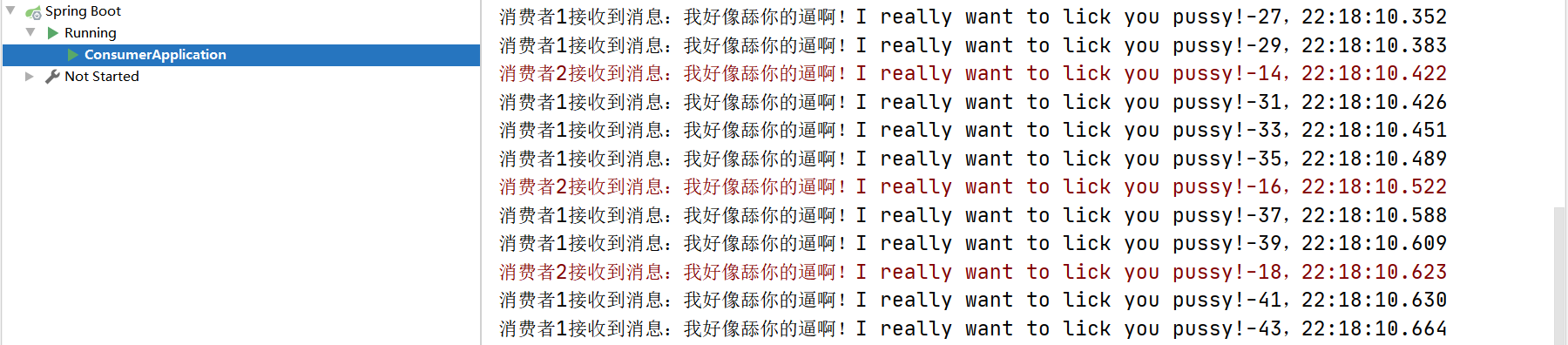
## 接收、消费消息

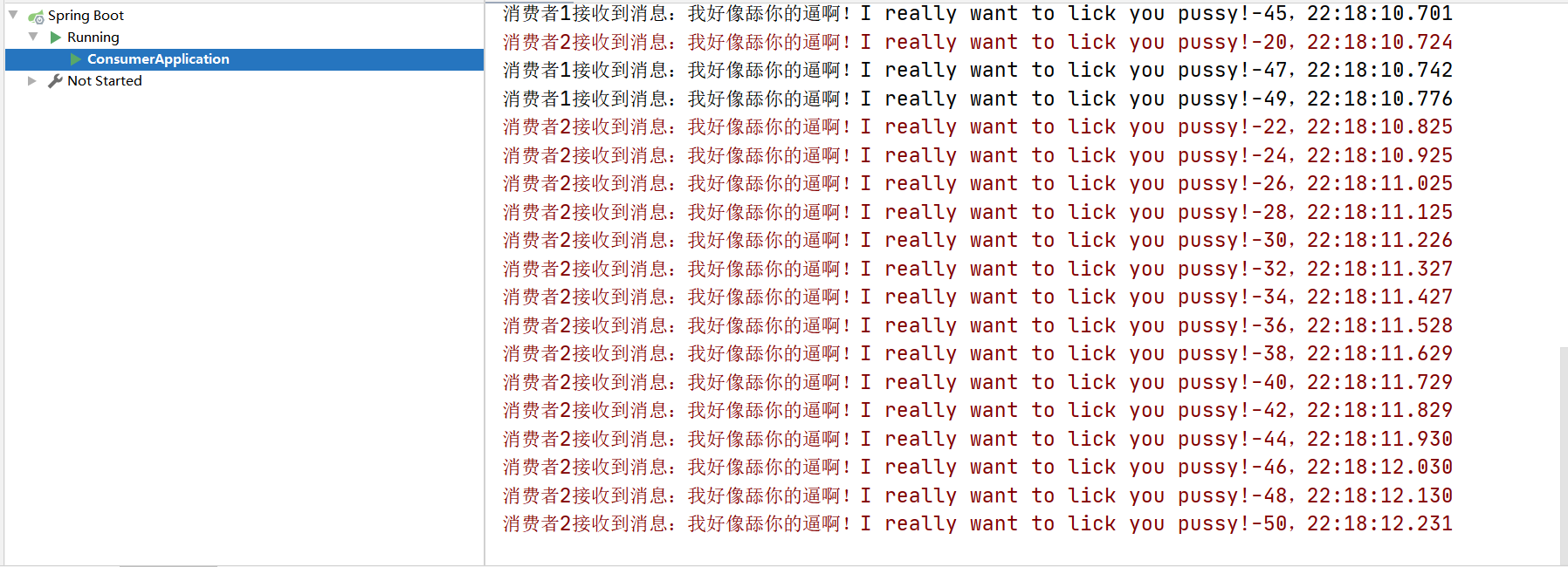
### 定义监听消息的类和接收、消费消息的方法



### 启动Consumer，查看控制台日志







#### 发现问题

查看日志可以发现，消息一没有按照消息发送的顺序消费的，二消息的消费时间耗时大概用了3秒！

不按顺序好理解，方法多线程执行占用CPU问题；

为什么时间耗时这么长呢？而且消费者1处理的都是奇数消息，消费者2处理的都是偶数消息，每个消费者接收处理的消费的数量是一样的！这显然不符合我们的预期！消费者1的处理能力是消费者2处理能力的5倍！显然消费者1应该多处理消息，能者多劳！然而事实却是消息平均分配给了两个消费者！这种分配方式没有考虑两个消费者处理消息的能力！

这是因为我们的RabbitMQ内部的机制造成的！

#### 消息预取机制

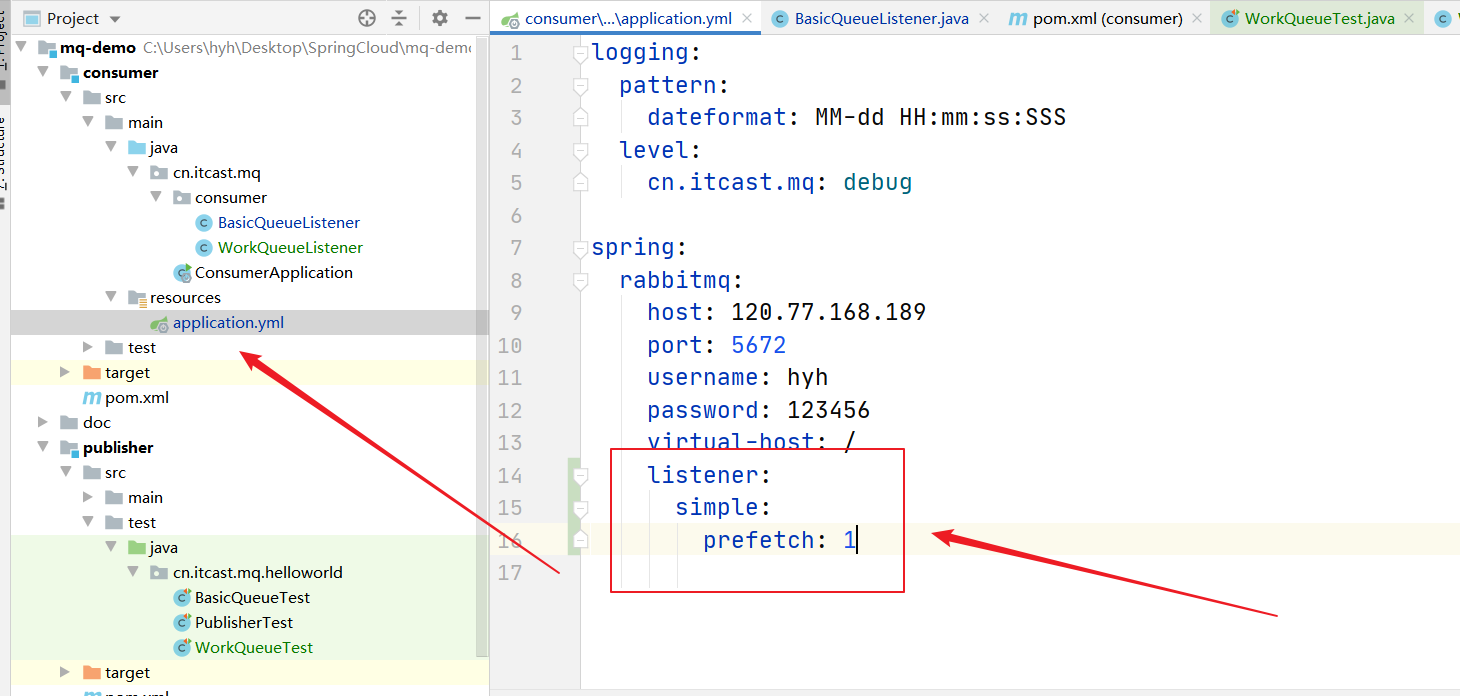
当有大量的消息到达队列时，队列会把消息进行投递，各个consumer的通道Channel会提前把消息拿过来，这就是消息预取！

每个consumer的通道Channel是轮流拿消息，不管对应的Consumer能不能处理，先拿过来再说，**于是各个consumer就会平均分配所有的消息**。但是consumer处理消息速度有快慢，有的consumer处理消息快一点，有的consumer处理消息慢一点，**于是这样子会导致处理消息的时间变长**。

要解决这个问题就是不能让Consumer的Channel拿太多消息，应该是当consumer没处理完一个消息，再去队列拿一个！没有金刚钻，别揽瓷器活！要配置：listener.simple.prefetch=1（每次只能获取一条消息，处理完成才能获取下一个消息）

#### 限制消息预取机制

修改application.yml配置文件，设置preFetch这个值，可以控制预消息的上限，默认值是无限，有多少拿多少，先拿过来再说！什么时候消费你别管，先拿过来！这显然不行，这个值一般要去做一些控制，看你的消费能力，消费能力差控制得低一点，消费能力高调得高一点，设成1就是每次取一条，消费完了你再来拿，这样就不会存在娶了一堆消息又处理不过来的情况！



重启consumer，看到问题解决，就不截图了好吗！