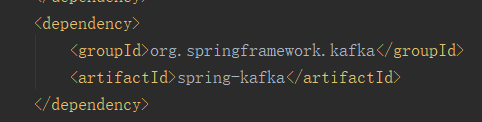
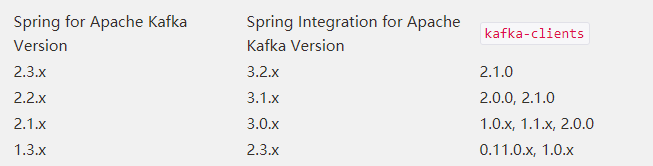
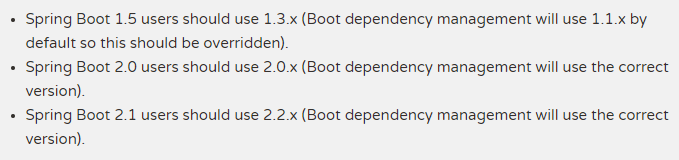
1. 在pom.xml中引入spring-kafka依赖：



注意：

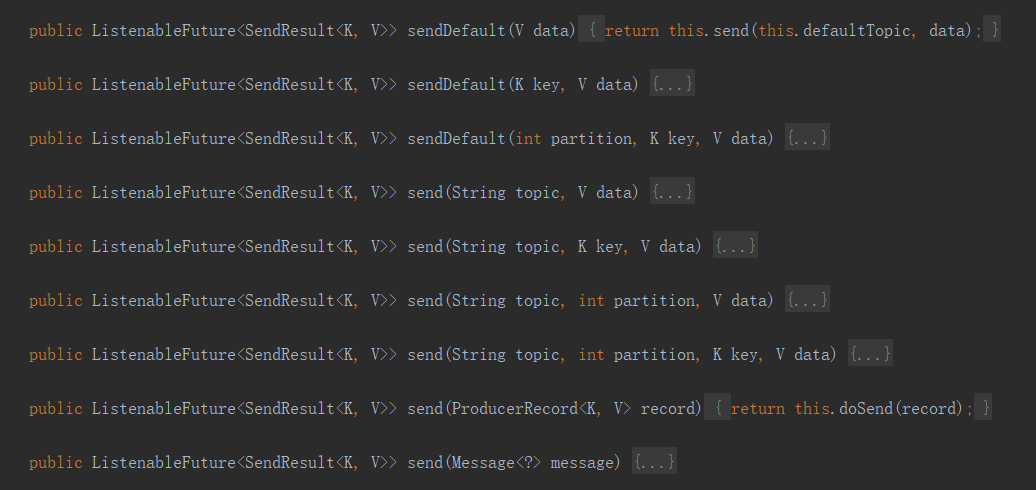
spring-kafka实际上也是对apache的kafka-client进行了包装和开发，所以使用的时候一定注意，引入的spring-kafka里封装的kafka-client的版本要和服务器上的kafka服务器版本要对应，不然就会产生问题，比如消费失败。对应版本如下：





具体可参考官方文档：https://spring.io/projects/spring-kafka#overview

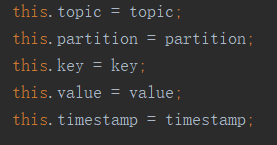
1. 消息发送--KafkaTemplate:
2. 该类简化了kafka中的消息发送操作，有以下send()方法：



各参数含义：

1. topic：这里填写的是Topic的名字
2. partition：这里填写的是分区的id，其实也是就第几个分区，id从0开始。表示指定发送到该分区中
3. timestamp：时间戳，一般默认当前时间戳
4. key：消息的键
5. data：消息的数据
6. ProducerRecord：消息对应的封装类，包含上述字段
7. Message<?>：Spring自带的Message封装类，包含消息及消息
8. 关于ProduceRecord类：

内部数据结构：



数据发送逻辑：

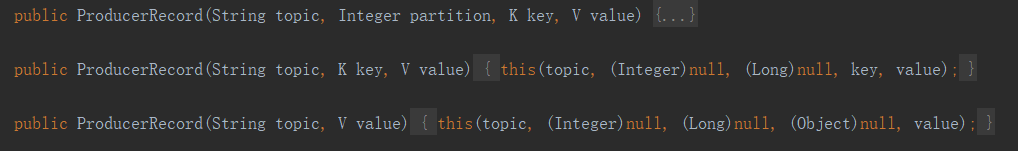
<1> 若指定Partition,则数据被发送至指定Partition

<2> 若未指定Partition,但指定了Key, 数据会按照hasy(key)发送至对应Partition

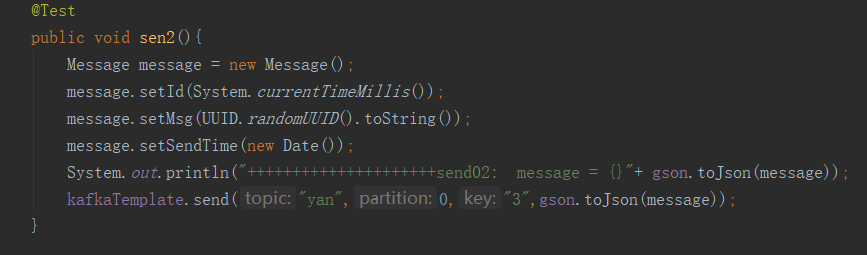
<3> 若既未指定Partition也没指定Key，数据会按照round-robin模式发送到每个Partition

<4> 若同时指定了Partition和Key, 数据只会发送到指定的Partition (Key不起作用，代码逻辑决定)

数据发送实现：



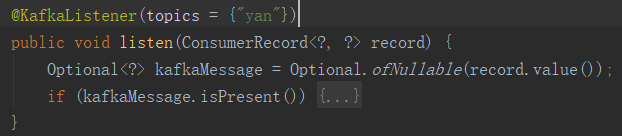
1. 注意：send() 方法默认则是异步的，若要同步只需要调用它的 get() 方法即可同步获取发送结果
2. 简单示例：



1. 消息监听
2. 使用@KafkaListener来注解实现监听的方法
3. 监听方法可接收参数：
4. Data:对于data值的类型其实并没有限定，根据KafkaTemplate所定义的类型来决定。data为List集合的则是用作批量消费。
5. ConsumerRecord：具体消费数据类，包含Headers信息、分区信息、时间戳等
6. Acknowledgment：用作Ack机制的接口
7. Consumer：消费者类，使用该类我们可以手动提交偏移量、控制消费速率等功能
8. @KafkaListner可添加属性：
9. id：消费者的id，当GroupId没有被配置的时候，默认id为GroupId
10. containerFactory：上面提到了@KafkaListener区分单数据还是多数据消费只需要配置一下注解的containerFactory属性就可以了，这里面配置的是监听容器工厂，也就是ConcurrentKafkaListenerContainerFactory，配置BeanName
11. topics：需要监听的Topic，可监听多个
12. topicPartitions：可配置更加详细的监听信息，必须监听某个Topic中的指定分区，或者从offset为200的偏移量开始监听
13. 使用ConsumerRecord类消费优点:

ConsumerRecord类里面包含分区信息、消息头、消息体等内容，如果业务需要获取这些参数时，使用ConsumerRecord会是个不错的选择

简单示例:



1. Ack机制确认消费：

与rabbitmq不同，Kafka是通过最新保存偏移量进行消息消费的，而且确认消费的消息并不会立刻删除，所以我们可以重复的消费未被删除的数据，当第一条消息未被确认，而第二条消息被确认的时候，Kafka会保存第二条消息的偏移量，也就是说第一条消息再也不会被监听器所获取，除非是根据第一条消息的偏移量手动获取。

使用Kafka的Ack机制比较简单，只需简单的三步即可：

1. 设置ENABLE\_AUTO\_COMMIT\_CONFIG=false，禁止自动提交
2. 设置AckMode=MANUAL\_IMMEDIATE
3. 监听方法加入Acknowledgment ack 参数

确认消息只需添加ack.acknowledge()方法，拒绝消息只需要注释掉上述方法即可。

今日：springboot整合kafka的消息生产、消费等操作

明日：整合这三周来的学习资料