1. 什么是消息中间件MOM？英文回答

As a middle tier software, message middleware creates, sends and receives messages in distributed systems.It provides a set of reliable and general methods to realize reliable, efficient and real-time in distributed system

Cross platform data transmission.

1. 描述MOM的主要应用场景。

答：面对规模和复杂度都越来越高的分布式系统，消息中间件技术显示出了它的优越性：①采用异步通信模式 ②客户和服务对象生命周期的松耦合关系异步消息处理

可以使用用于一些场景：

①异步：可以将一些非核心流程，如日志，短信，邮件等，通过MQ的方式异步去处理。这样做的好处是缩短主流程的响应时间，提升用户体验。

1. 应用解耦合

商品服务和订单服务之间。用户下单后，订单服务会通知商品服务。不使用MQ的情况是订单服务调用商品服务的接口，这样订单服务和商品服务之间是耦合的；使用MQ，订单服务完成持久化处理，将消息写入MQ消息队列中，返回用户订单下单成功，商品服务来订阅这个下单的消息，采用拉或推的方式获得下单信息，商品服务根据商品下单信息进行商品库存信息修改，这样当下单时商品服务不可用时，也不影响正常下单，这就完成了订单服务和商品服务之间的解耦。

1. 流量消峰

秒杀时流量过大，导致流量暴增，最终可能导致应用挂掉。一般会在应用前端加入消息队列来控制活动人数，假如消息队列超过最大数量，应该直接抛弃用户请求或者跳转到错误页面。秒杀根据消息队列中的请求信息在做后续的业务处理。比如在抢购时，可能一下子过来了10万个请求，但MQ只接受前100个用户的请求，超过100个不接收了。这样就成功限制了用户请求。

1. 认真阅读《分布式消息中间件实践.pdf》第一章消息队列1.4，请设计一个自己的消息队列系统。

要求：1.请写出三种Java创建线程的方式。

2.简单说明消息服务中Broker的作用。

3.针对《分布式消息中间件实践.pdf》中消息队列示例，你认为那些方面还可以改进？

4.参考消息队列示例，设计并实现MyFirstMQ。

3.1

继承Thread类创建

　通过继承Thread并且重写其run()，run方法中即线程执行任务。创建后的子类通过调用 start() 方法即可执行线程方法。通过继承Thread实现的线程类，多个线程间无法共享线程类的实例变量。（需要创建不同Thread对象，自然不共享）

2、

通过Runnable接口创建线程类

　该方法需要先 定义一个类实现Runnable接口，并重写该接口的 run() 方法，此run方法是线程执行体。接着创建 Runnable实现类的对象，作为创建Thread对象的参数target，此Thread对象才是真正的线程对象。通过实现Runnable接口的线程类，是互相共享资源的。

3.

使用Callable和Future创建线程

　　从继承Thread类和实现Runnable接口可以看出，上述两种方法都不能有返回值，且不能声明抛出异常。而Callable接口则实现了此两点，Callable接口如同Runable接口的升级版，其提供的call()方法将作为线程的执行体，同时允许有返回值。但是Callable对象不能直接作为Thread对象的target，因为Callable接口是 Java 5 新增的接口，不是Runnable接口的子接口。对于这个问题的解决方案，就引入 Future接口，此接口可以接受call() 的返回值，RunnableFuture接口是Future接口和Runnable接口的子接口，可以作为Thread对象的target 。并且， Future 接口提供了一个实现类：FutureTask 。FutureTask实现了RunnableFuture接口，可以作为 Thread对象的target。

3.2简单说明消息服务中Broker的作用。

broker是一个单独的服务,提供消息的转发和存储功能

3.3针对《分布式消息中间件实践.pdf》中消息队列示例，你认为那些方面还可以改进？

答：

a.关于消息堆积:消息在处理中心逐渐积压而得不到释放。比如给消息队列设置一个阈值,将超过阈值的消息不再放入处理中心,以防止系统资源被耗尽,导致机器挂掉甚至整个消息队列不可用。

b.消息持久化: 将消息放在内存中存在的最大问题是,一旦机器宿掉消息将丢失。持久化方案有很多种,比如将消息存到本地文件、分布式文件系统、数据库系统中等。

可靠投递:可靠投递是不允许存在消息丢失的情况的。从消息的整个生命周期来分析,消息丢失的情况一般发生在如下过程中:

从生产者到消息处理中心。

从消息处理中心到消息消费者。

消息处理中心持久化消息。

c.消息重复:为了支持消息可靠投递,当消息发送失败或者不知道是否发送成功时(比如超时),消息的状态是待发送,定时任务不停地轮询所有的待发送消息,最终保证消息不会丢失,这就带来了消息可能会重复的问题。

3.4参考消息队列示例，设计并实现MyFirstMQ

见java文件