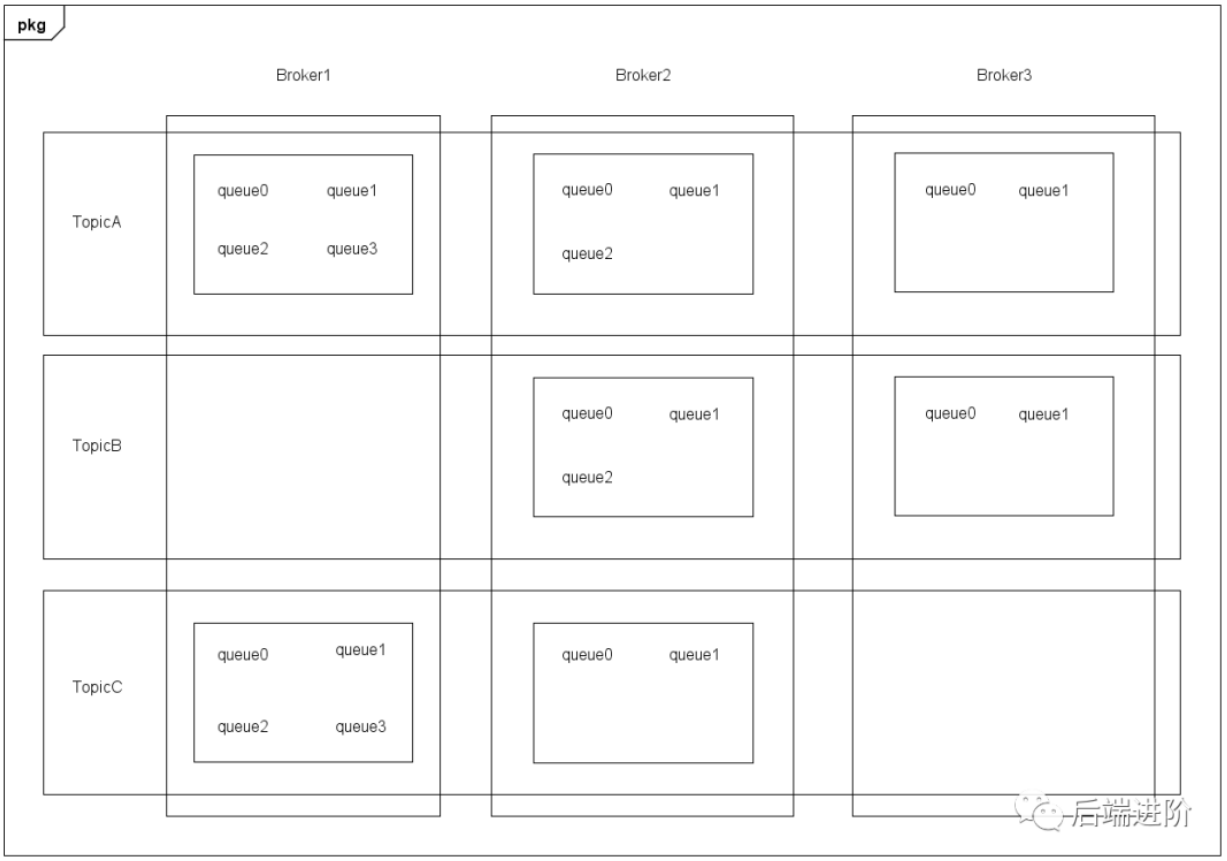
RocketMQ五种消息类型

RocketMQ支持五种消息类型（后面会有独立的案例）

- **普通消息：普通消息也称为并发消息，和传统的队列相比，并发消息没有顺序，但是生产消费都是并行进行的，单机性能可达十万级别的TPS。**
- **分区有序消息：与Kafka中的分区类似，把一个Topic消息分为多个分区“保存”和消费，在一个分区内的消息就是传统的队列，遵循FIFO（先进先出）原则。**
- **全局有序消息：如果把一个 Topic 的分区数设置为 1，那么该 Topic 中的消息就是单分区，所有消息都遵循FIFO（先进先出）的原则。**
- **延迟消息：消息发送后，消费者要在一定时间后，或者指定某个时间点才可以消费。在没有延迟消息时，基本的做法是基于定时计划任务调度，定时发送消息。在RocketMQ中只需要在发送消息时设置延迟级别即可实现。**
- **事务消息：主要涉及分布式事务，即需要保证在多个操作同时成功或者同时失败时，消费者才能消费消息。RocketMQ通过发送Half消息、处理本地事务、提交（Commit）消息或者回滚（Rollback）消息优雅地实现分布式事务。**

消息类型	优　点	缺　点	备　注
普通消息（并发消息）	性能最好。单机 TPS 的级别为 100 000	消息的生产和消费都无序	大部分场景适用
分区有序消息	单分区中消息有序，单机发送 TPS 万级别	单点问题。如果 Broker 宕机，则会导致发送失败	大部分有序消息场景适用
全局有序消息	类似传统的 Queue，全部消息有序，单机发送 TPS 千级别	单点问题。如果 Broker 宕机，则会导致发送失败	极少场景使用
延迟消息	RocketMQ 自身支持，不需要额外使用组件，支持延迟特性	不能根据任意时间延迟，使用范围受限。Broker 随着延迟级别增大支持越多，CPU 压力越大，延迟时间不准确	非精确、延迟级别不多的场景，非常方便使用
事务消息	RocketMQ 自身支持，不需要额外使用组建支持事务特性	RocketMQ 事务是生产者事务，只有生产者参与，如果消费者处理失败则事务失效	简单事务处理可以使用

Broker、分区、队列的关系


pkg
Broker1
Broker2
Broker3
TopicA
TopicB
TopicC
queue0
queue1
queue2
queue3
后端进阶