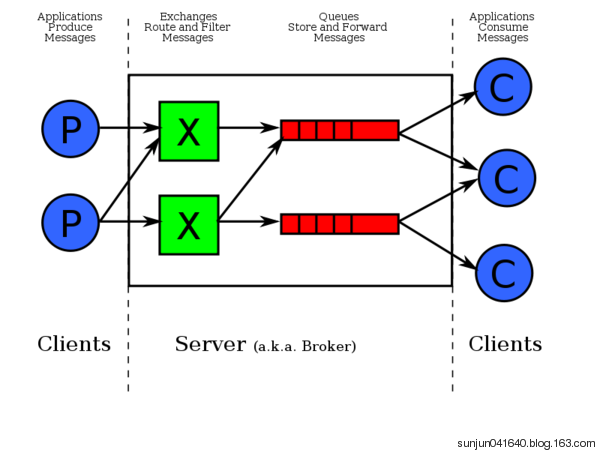
# RabbitMQ 实现原理

# AMQP(高级消息队列协议 Advanced Message Queue Protocol)

AMQP当中有四个概念非常重要: 虚拟主机（virtual host），交换机（exchange），队列（queue）和绑定（binding）。一个虚拟主机持有一组交换机、队列和绑定。为什么需要多个虚拟主机呢？很简单，RabbitMQ当中，用户只能在虚拟主机的粒度进行权限控制。因此，如果需要禁止A组访问B组的交换机/队列/绑定，必须为A和B分别创 建一个虚拟主机。每一个RabbitMQ服务器都有一个默认的虚拟主机“/”。

Producer 要产生消息必须要创建一个 Exchange ，Exchange 用于转发消息，但是它不会做存储，如果没有 Queue bind 到 Exchange 的话，它会直接丢弃掉 Producer 发送过来的消息，当然如果消息总是发送过去就被直接丢弃那就没有什么意思了，一个 Consumer 想要接受消息的话，就要创建一个 Queue ，并把这个 Queue bind 到指定的 Exchange 上，然后 Exchange 会把消息转发到 Queue 那里，Queue 会负责存储消息，Consumer 可以通过主动 Pop 或者是 Subscribe 之后被动回调的方式来从 Queue 中取得消息。

**Exchange，Queue，RoutingKey**



蓝色-- Client（相对于Rabbitmq Server来说）

绿色--Exchange

红色—Queue

     - 交换器（Exchange），它是发送消息的实体。

     - 队列（Queue），这是接收消息的实体。

     - 绑定器（Bind），将交换器和队列连接起来，并且封装消息的路由信息。

Exchange指向Queue的黑色线—RoutingKey，可以将它简单的理解为一条连接Exchange和Queue的路线

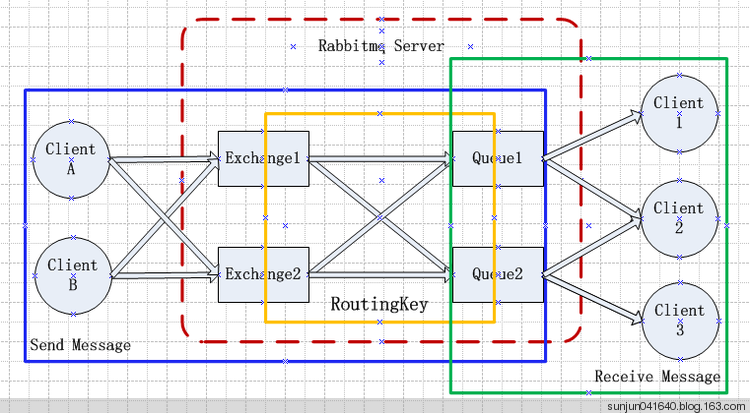
Exchange和Queue都需要通过channel来进行定义，而RoutingKey则只需要在binding时取个名字就行了。

这一块的理解是不正确的，

Exchange Queue RoutingKey关系说明：

|  |  |  |
| --- | --- | --- |
| **Exchange Name** | **Queue Name** | **Routing Key** |
|  | test.queue | test.queue |
|  | test.queue2 | test.queue2 |
| test.exchange | test.queue | test.routingkey |
| test.exchange | test.queue2 | test.routingkey |
| test.exchange | test.queue | test.routingkey2 |
| test.exchange1 | test.queue | test.routingkey |

由结果可以看出，由Exchange，Queue，RoutingKey三个才能决定一个从Exchange到Queue的唯一的线路。



左边的Client向右边的Client发送消息，流程：

1，  获取Conection

2，  获取Channel

3，  定义Exchange，Queue

4，  使用一个RoutingKey将Queue Binding到一个Exchange上

5，  通过指定一个Exchange和一个RoutingKey来将消息发送到对应的Queue上，

6，  接收方在接收时也是获取connection，接着获取channel，然后指定一个Queue直接到它关心的Queue上取消息，它对Exchange，RoutingKey及如何binding都不关心，到对应的Queue上去取消息就OK了

一个Client发送消息，哪些Client可以收到消息，其核心就在于Exchange，RoutingKey，Queue的关系上。

|  |  |  |  |
| --- | --- | --- | --- |
|  | Exchange | RoutingKey | Queue |
| 1 | E1 | R1 | Q1 |
| 2 |  | R2 | Q2 |
| 3 | E2 | R3 | Q1 |
| 4 |  | R4 | Q2 |
| 5 | E1 | R5 | Q1 |
| 6 | E2 | R6 | Q1 |

我们可以这样理解，RoutingKey就像是个中间表，将两个表的数据进行多对多关联，只不过对于相同的Exchange和Queue，可以使用不同的RoutingKey重复关联多次。