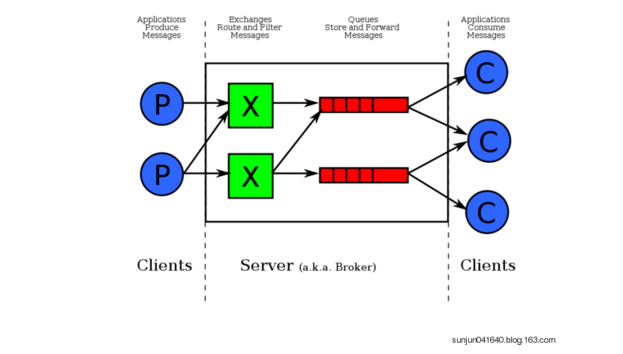
# 基本介绍

## MQ消息队列

* 1. 定义

1. 消息队列（Message Queue，简称MQ），从字面意思上看，本质是个队列，FIFO先入先出，只不过队列中存放的内容是message而已。
2. 其主要用途：不同进程Process/线程Thread之间通信。
   1. 应用背景
3. 不同进程（process）之间传递消息时，两个进程之间耦合程度过高，改动一个进程，引发必须修改另一个进程，为了隔离这两个进程，在两进程间抽离出一层（一个模块），所有两进程之间传递的消息，都必须通过消息队列来传递，单独修改某一个进程，不会影响另一个；
4. 不同进程（process）之间传递消息时，为了实现标准化，将消息的格式规范化了，并且，某一个进程接受的消息太多，一下子无法处理完，并且也有先后顺序，必须对收到的消息进行排队，因此诞生了事实上的消息队列；
5. MQ框架非常之多，比较流行的有RabbitMq、ActiveMq、ZeroMq、kafka，以及阿里开源的RocketMQ。本文主要介绍RabbitMq。
   1. MQ服务的具体实现



1. 队列服务通常有三个概念： 发消息者（生产者）、队列、收消息者（消费者）。

RabbitMQ 在这个基本概念之上, 多做了一层抽象, 在发消息者（生产者）和队列之间, 加入了交换器 (Exchange). 这样发消息者和队列就没有直接联系, 转而变成发消息者把消息给交换器, 交换器根据调度策略再把消息再给队列。

1. 左侧 P 代表 生产者，也就是往 RabbitMQ 发消息的程序。
2. 中间即是 RabbitMQ，其中包括了 交换机 和 队列。
3. 右侧 C 代表 消费者，也就是往 RabbitMQ 拿消息的程序。
   * 1. 虚拟主机
4. 一个虚拟主机持有一组交换机、队列和绑定。
5. 为什么需要多个虚拟主机呢？

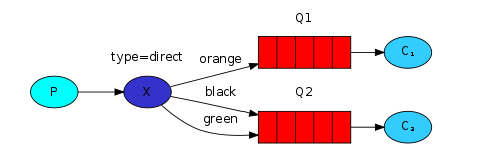
RabbitMQ当中，用户只能在虚拟主机的粒度进行权限控制。 因此，如果需要禁止A组访问B组的交换机/队列/绑定，必须为A和B分别创建一个虚拟主机。每一个RabbitMQ服务器都有一个默认的虚拟主机“/”。

* + 1. 交换机（Exchange）

1. 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。 这里有一个比较重要的概念：路由键 。消息到交换机的时候，交互机会转发到对应的队列中，那么究竟转发到哪个队列，就要根据该路由键。
   * 1. 绑定
2. 绑定就是交换机需要和队列相绑定，这其中如上图所示，是多对多的关系。
   1. MQ中的交换机

交换机的功能主要是**接收消息并且转发到绑定的队列**，交换机不存储消息，在启用ack模式后，交换机找不到队列会返回错误。主要有四种：

1. Direct：direct 类型的行为是”先匹配, 再投送”. 即在绑定时设定一个 routing\_key, 消息的routing\_key 匹配时, 才会被交换器投送到绑定的队列中去.
2. Topic：按规则转发消息（最灵活）
3. Headers：设置header attribute参数类型的交换机
4. Fanout：转发消息到所有绑定队列
   * 1. Direct Exchange



1. Direct Exchange是RabbitMQ默认的交换机模式，也是最简单的模式，根据key全文匹配去寻找队列。
2. 第一个 X - Q1 就有一个 binding key，名字为 orange； X - Q2 就有 2 个 binding key，名字为 black 和 green。当消息中的 路由键 和 这个 binding key 对应上的时候，那么就知道了该消息去到哪一个队列中。
   * 1. Topic Exchange
3. Topic Exchange 转发消息主要是根据通配符。 在这种交换机下，队列和交换机的绑定会定义一种路由模式，那么，通配符就要在这种路由模式和路由键之间匹配后交换机才能转发消息。
4. 路由键必须是一串字符，用句号（.）隔开，比如说 agreements.us，或者

agreements.eu.stockholm 等。

1. 路由模式必须包含一个\*号，主要用于匹配路由键指定位置的一个单词。比如说，一个路由模式是：agreements..b.\*，那么匹配路由键就必须第一个单词为agreements,第三个为b,第四个单词任意但只能有四个单词。
2. 井号（#）就表示相当于一个或者多个单词，例如一个匹配模式是agreements.eu.berlin.#，那么，以agreements.eu.berlin开头的路由键都是可以

的。

1. 发送格式



* + 1. Headers Exchange\

1. headers 也是根据规则匹配, 相较于 direct 和 topic 固定地使用 routing\_key , headers 则是一个自定义匹配规则的类型. **在队列与交换器绑定时, 会设定一组键值对规则, 消息中也包括一组键值对( headers 属性)**, 当这些键值对有一对, 或全部匹配时, 消息被投送到对应队列.
   * 1. Fanout Exchange
2. Fanout Exchange 消息广播的模式，不管路由键或者是路由模式，会把消息发给绑定给它的全部队列，如果配置了routing\_key会被忽略。

## RabbitMQ

* 1. 简介

1. 消息队列是应用程序和应用程序之间的通讯方法。
2. RabbitMQ是一个开源的，在AMQP基础上完整的，可复用的企业消息系统。
3. 支持多个操作系统，多种语言开发。

## AMQP

1. AMQP，即Advanced Message Queuing Protocol,**一个提供统一消息服务的应用层标准高级消息队列协议**,是应用层协议的一个开放标准,为面向消息的中间件设计。基于此协议的客户端与消息中间件可传递消息，并不受客户端/中间件不同产品，不同的开发语言等条件的限制。Erlang中的实现有 RabbitMQ等。

# 环境搭建

## 底层语言支持-ErLang

1. 官网下载，exe安装
2. 配置环境变量 ERLANG\_HOME = 安装目录

## RabbitMQ安装

1. 官网下载，exe安装
2. 安装管理工具

\sbin> rabbitmq-plugins enable rabbitmq\_management

1. RabbitMQ服务是自动安装的，不用手动添加
2. 登录127.0.0.1:15672配置用户