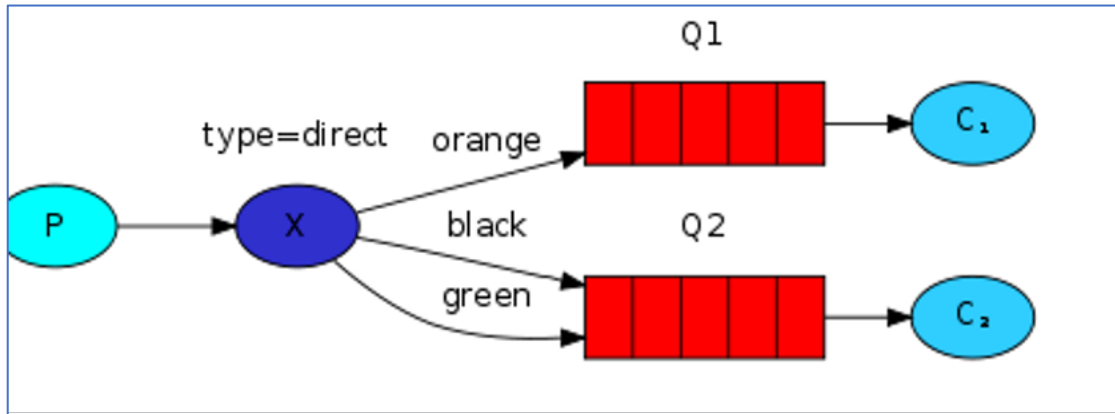


# RabbitMQ 面试题

## 1、RabbitMQ routing 路由模式



1、消息生产者将消息发送给交换机按照路由判断,路由是字符串(info) 当前产生的消息携带路由字符(对象的方法), 交换机根据路由的 key, 只能匹配上路由 key 对应的消息队列, 对应的消费者才能消费消息。

2、根据业务功能定义路由字符串。

3、从系统的代码逻辑中获取对应的功能字符串,将消息任务扔到对应的队列中。

4、业务场景 : error 通知、EXCEPTION、错误通知的功能、传统意义的错误通知、客户通知、利用 key 路由, 可以将程序中的错误封装成消息传入到消息队列中, 开发者可以自定义消费者, 实时接收错误。

## 2、消息怎么路由？

消息提供方->路由->一至多个队列消息发布到交换器时, 消息将拥有一个路由键

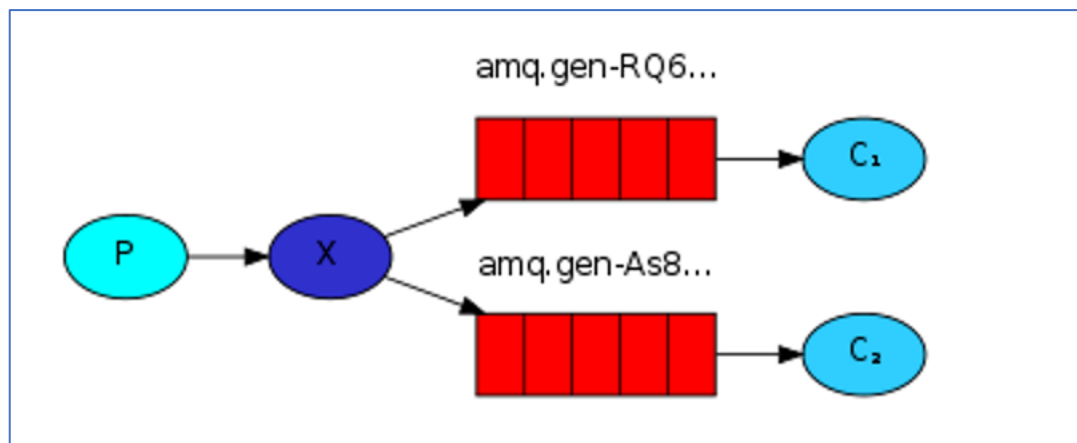
( routing key ), 在消息创建时设定。通过队列路由键, 可以把队列绑定到交换器上。消息到达交换器后, RabbitMQ 会将消息的路由键与队列的路由键进行匹配 ( 针对不同的交

换器有不同的路由规则 )。

常用的交换器主要分为一下三种：

- fanout：如果交换器收到消息，将会广播到所有绑定的队列上。
- direct：如果路由键完全匹配，消息就被投递到相应的队列。
- topic：可以使来自不同源头的消息能够到达同一个队列。使用 topic 交换器时，可以使用通配符。

### 3、RabbitMQ publish/subscribe 发布订阅(共享资源)



- 每个消费者监听自己的队列。
- 生产者将消息发给 broker，由交换机将消息转发到绑定此交换机的每个队列，每个绑定交换机的队列都将接收到消息。

### 4、能够在地理上分开的不同数据中心使用 RabbitMQ cluster 么？

不能。

第一，你无法控制所创建的 queue 实际分布在 cluster 里的哪个 node 上（一般使用

HAProxy + cluster 模型时都是这样), 这可能会导致各种跨地域访问时的常见问题。

第二, Erlang 的 OTP 通信框架对延迟的容忍度有限, 这可能会触发各种超时, 导致业务疲于处理。

第三, 在广域网上的连接失效问题将导致经典的“脑裂”问题, 而 RabbitMQ 目前无法处理 (该问题主要是说 Mnesia)。

## 5、RabbitMQ 有那些基本概念？

- Broker : 简单来说就是消息队列服务器实体。
- Exchange : 消息交换机, 它指定消息按什么规则, 路由到哪个队列。
- Queue : 消息队列载体, 每个消息都会被投入到一个或多个队列。
- Binding : 绑定, 它的作用就是把 exchange 和 queue 按照路由规则绑定起来。
- Routing Key : 路由关键字, exchange 根据这个关键字进行消息投递。
- VHost : vhost 可以理解为虚拟 broker , 即 mini-RabbitMQ server。其内部均含有独立的 queue、exchange 和 binding 等, 但最最重要的是, 其拥有独立的权限系统, 可以做到 vhost 范围的用户控制。当然, 从 RabbitMQ 的全局角度, vhost 可以作为不同权限隔离的手段 (一个典型的例子就是不同的应用可以跑在不同的 vhost 中)。
- Producer : 消息生产者, 就是投递消息的程序。
- Consumer : 消息消费者, 就是接受消息的程序。
- Channel : 消息通道, 在客户端的每个连接里, 可建立多个 channel, 每个 channel 代表一个会话任务。

- 由 Exchange、Queue、RoutingKey 三个才能决定一个从 Exchange 到 Queue 的唯一的线路。

## 6、什么情况下会出现 blackholed 问题？

blackholed 问题是指，向 exchange 投递了 message，而由于各种原因导致该 message 丢失，但发送者却不知道。可导致 blackholed 的情况：1.向未绑定 queue 的 exchange 发送 message；2.exchange 以 binding\_key key\_A 绑定了 queue queue\_A，但向该 exchange 发送 message 使用的 routing\_key 却是 key\_B。

## 7、什么是消费者 Consumer？

消费消息，也就是接收消息的一方。

消费者连接到 RabbitMQ 服务器，并订阅到队列上。消费消息时只消费消息体，丢弃标签。

## 8、消息如何分发？

- 若该队列至少有一个消费者订阅，消息将以循环（round-robin）的方式发送给消费者。每条消息只会分发给一个订阅的消费者（前提是消费者能够正常处理消息并进行确认）。

- 通过路由可实现多消费的功能

## 9、Basic.Reject 的用法是什么？

该信令可用于 consumer 对收到的 message 进行 reject 。若在该信令中设置 `requeue=true` , 则当 RabbitMQ server 收到该拒绝信令后, 会将该 message 重新发送到下一个处于 consume 状态的 consumer 处 ( 理论上仍可能将该消息发送给当前 consumer )。若设置 `requeue=false` , 则 RabbitMQ server 在收到拒绝信令后, 将直接将该 message 从 queue 中移除。

另外一种移除 queue 中 message 的小技巧是, consumer 回复 Basic.Ack 但不对其获取到的 message 做任何处理。而 Basic.Nack 是对 Basic.Reject 的扩展, 以支持一次拒绝多条 message 的能力。

## 10、什么是 Binding 绑定？

通过绑定将交换器和队列关联起来, 一般会指定一个 BindingKey, 这样 RabbitMq 就知道如何正确路由消息到队列了。