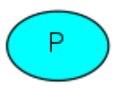
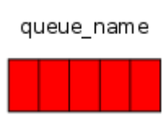
Python RabbitMQ基础

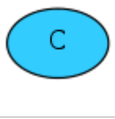
RabbitMQ是一个消息代理，它的工作就是接收和转发消息，你可以把它想像成一个邮局：你把信件放入邮箱，邮递员就会把信件投递到你的收件人处。在这个比喻中，RabbitMQ就扮演着邮箱、邮局以及邮递员的角色。下面是RabbitMQ和消息所涉及到的一些术语，生产(Producing)的意思就是发送，发送消息的程序就是一个生产者(producer)，我们一般用"P"来表示：



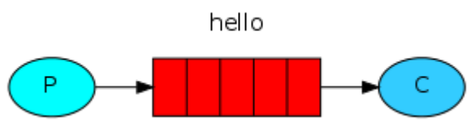
队列(queue)就是存在于RabbitMQ中邮箱的名称，虽然消息的传输经过了RabbitMQ和应用程序，但是它只能被存储于队列当中。实质上队列就是个巨大的消息缓冲区，它的大小只受主机内存和硬盘限制。多个生产者(producers)可以把消息发送给同一个队列，同样多个消费者(consumers)也能够从同一个队列(queue)中获取数据。



在这里消费(Consuming)和接收(receiving)是同一个意思，一个消费者(consumer)就是一个等待获取消息的程序，我们把它绘制为"C"：



需要指出的是生产者、消费者、代理需不要待在同一个设备上，事实上大多数应用也确实不在会将他们放在一台机器上，接下来我们用Python写两个小程序。一个发送单条消息的生产者(producer)和一个接收消息并将其输出的消费者(consumer)，传递的消息是"Hello World".下图中"P"代表生产者，"C"代表消费者，中间的盒子代表为消费者保留的消息缓冲区，也就是我们的队列。



生产者(producer)把消息发送到一个名为"hello"的队列中，消费者(consumer)从这个队列中获取消息。

示例代码：

In [ ]:
```
import pika
credentials = pika.PlainCredentials('rabbit', 'rabbit')
connection = pika.BlockingConnection(pika.ConnectionParameters('172.16.70.251', 5672,'/',credentials))
channel = connection.channel()
channel.queue_declare(queue='hello')
channel.basic_publish(exchange='',
                      routing_key='hello',
                      body='Hello World!')
print(" [x] Sent 'Hello World!'")
connection.close()
```

示例代码：

In [ ]:
```
import pika
credentials = pika.PlainCredentials('rabbit', 'rabbit')
connection = pika.BlockingConnection(pika.ConnectionParameters('172.16.70.251', 5672,'/',credentials))
channel = connection.channel()
channel.queue_declare(queue='hello')
def callback(ch, method, properties, body):
    print(" [x] Received %r" % body)
channel.basic_consume('hello', callback, auto_ack=True)
print(' [*] Waiting for messages. To exit press CTRL+C')
channel.start_consuming()
```