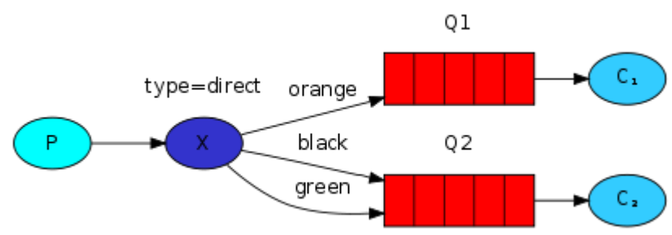
Python RabbitMQ路由

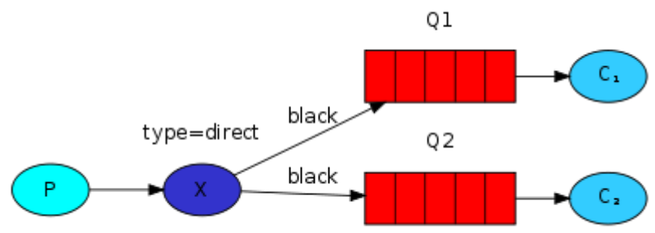
在前面的例子中我们使用在前面的教程中，我们使用扇形交换机，把日志消息广播给多个接收者．扇型交换机的缺点是不够的灵活性因为它能做的仅仅是广播．
在本例中我们将会使用直连交换机，路由的算法很简单，交换机将会对绑定键(binding key)和路由键(routing key)进行精确匹配，从而确定消息该分发到哪个队列．

下图能够很好的描述这个场景：



在这个场景中，我们可以看到直连交换机x和两个队列进行了绑定，第一个队列使用orange作为绑定键，第二个队列有两个绑定，一个使用black作为绑定键另外一个使用green．这样以来，当路由键为orange的消息发布到交换机，就会被路由到队列Q1．路由键为black或者green的消息就会路由到Q2，其他的所有消息都将会被丢弃．

多个绑定：



多个队列使用相同的绑定键是合法的，在上图场景中，我们可以添加一个x交换机和Q1 Q2的绑定，使用black绑定键．这样一来此时的直连交换机就和扇型交换机的行为一样，会将消息广播到所有匹配的队列．也就是带有black路由键的消息会同时发送到Q1和Q2．

示例代码(生产者)：

In [ ]:
```
#coding:utf8
import pika, sys
cert = pika.PlainCredentials('rabbit', 'rabbit')
para = pika.ConnectionParameters('172.16.70.251', '5672', '/', cert)
connect = pika.BlockingConnection(para)
channel = connect.channel()
channel.exchange_declare(exchange="direct_logs",
                         exchange_type="direct")
level = sys.argv[1]
msg = sys.argv[2]
channel.basic_publish(exchange='direct_logs',
                      routing_key=level,
                      body=msg
                      )
print('message level %s body %s send done!!' %(level, msg))
connect.close()
```

示例代码(消费者1)：

In [ ]:
```
#!/usr/bin/python
import pika, sys
import time
cert = pika.PlainCredentials('rabbit', 'rabbit')
para = pika.ConnectionParameters('172.16.70.251', '5672', '/', cert)
connect = pika.BlockingConnection(para)
channel = connect.channel()
channel.exchange_declare(exchange="direct_logs", exchange_type="direct")
result = channel.queue_declare(exclusive=True, queue='')
queue_name = result.method.queue
print(queue_name)
levels = ["black", "green"]
for level in levels:
   print(level)
   channel.queue_bind(exchange='direct_logs',
                      queue=queue_name,
                      routing_key=level)

def callback(ch, method, prop, body):
   print('this is callback level is %s args is %s'%(method.routing_key, body))
channel.basic_consume(queue_name, callback, auto_ack=True)
channel.start_consuming()
```

示例代码(消费者2)：

In [ ]:
```
#!/usr/bin/python
import pika, sys
import time
cert = pika.PlainCredentials('rabbit', 'rabbit')
para = pika.ConnectionParameters('172.16.70.251', '5672', '/', cert)
connect = pika.BlockingConnection(para)
channel = connect.channel()
channel.exchange_declare(exchange="direct_logs", exchange_type="direct")
result = channel.queue_declare(exclusive=True, queue='')
queue_name = result.method.queue
levels = ["orange"]
for level in levels:
   channel.queue_bind(exchange='direct_logs',
                      queue=queue_name,
                      routing_key=level)

def callback(ch, method, prop, body):
   print('this is callback level %s args is %s'%(method.routing_key, body))
channel.basic_consume(queue_name, callback, auto_ack=True)
channel.start_consuming()
```

实验效果：
在生产者上分别发送等级为Black Green Orange的信息，可知消费者1可以接受black 与 green，消费者2可以接受orange

In [ ]: