



链滴

# [free] 来自 cloudfamqp 的 rabbitmq 服务

作者: [alanfans](#)

原文链接: <https://ld246.com/article/1530198219168>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

## 来源广告邮件

**T** The CloudAMQP Team

⋮



28 Jun 2018, 18:28 ▾



# CloudAMQP

Start for FREE now!



Hello there,

A little fun fact for you!

Did you know that CloudAMQP has become the **industry leader in RabbitMQ** as a service **in only 4 years...?**

Some might call it luck. But we know it is due to our **commitment to excellence and customer satisfaction**. We want to take away the worry you might feel around configuring and setting up your own RabbitMQ server...

So that you can simply focus on **the core part of your applications!**

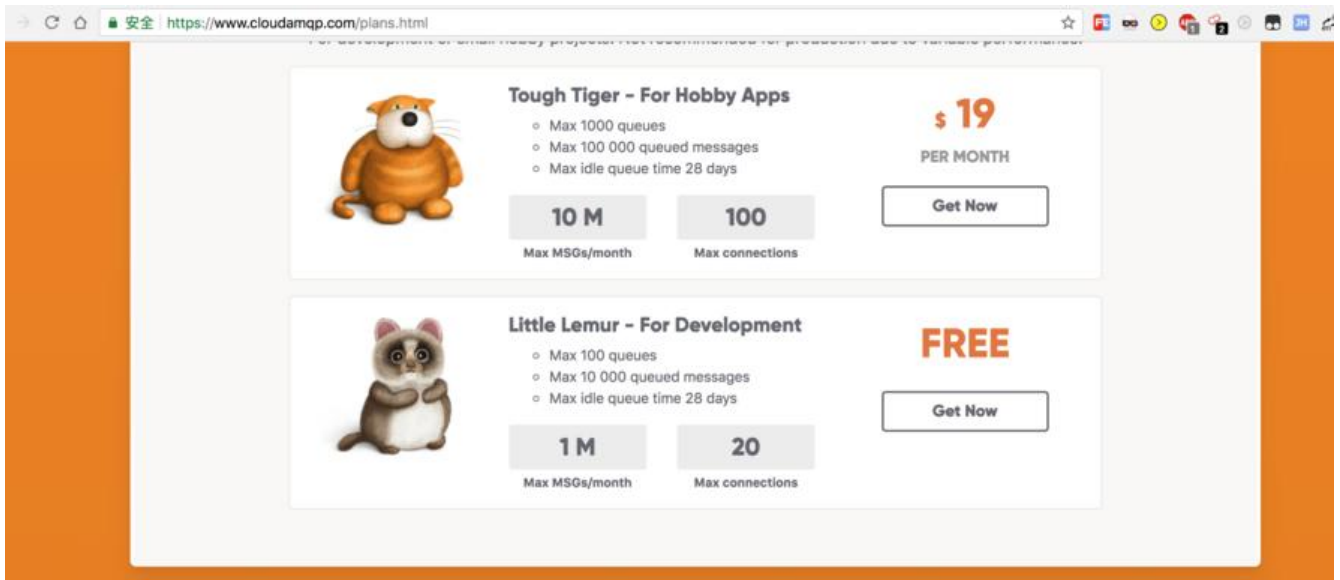
In fact... We are so dedicated to this that we have created a plan where you can start our service at no cost!

Do you want to try out our service and see if it is for you? Then sign up for our **FREE plan** now.

原文链接: [\[free\] 来自 cloudamqp 的 rabbitmq 服务](#)



For development or small hobby projects. Not recommended for production due to variable performance.---意思是测试用，不能用于生产

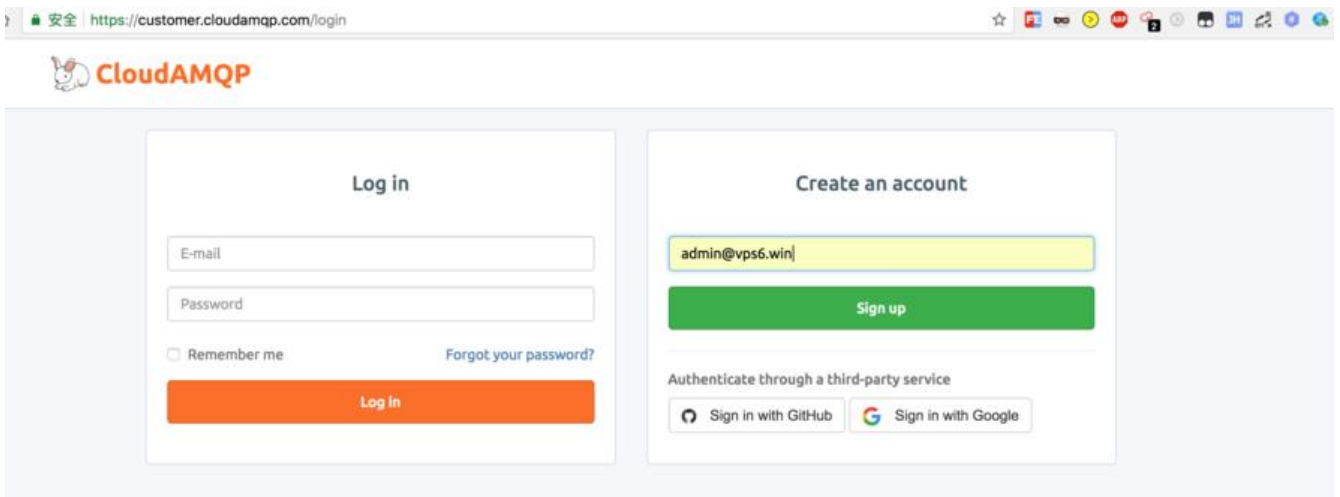


The screenshot shows the CloudAMQP pricing page with two plans:

- Tough Tiger - For Hobby Apps**
  - Max 1000 queues
  - Max 100 000 queued messages
  - Max idle queue time 28 days
  - Price: \$19 PER MONTH
  - Max MSGs/month: 10 M
  - Max connections: 100
- Little Lemur - For Development**
  - Max 100 queues
  - Max 10 000 queued messages
  - Max idle queue time 28 days
  - Price: FREE
  - Max MSGs/month: 1 M
  - Max connections: 20

## 注册走起

<https://www.cloudamqp.com/>



The screenshot shows the CloudAMQP login and registration page. The 'Log in' section includes fields for E-mail and Password, a 'Remember me' checkbox, a 'Forgot your password?' link, and a 'Log in' button. The 'Create an account' section includes a text input field with 'admin@vps6.win', a 'Sign up' button, and options to 'Authenticate through a third-party service' using GitHub or Google.

## Create an account

Welcome to CloudAMQP! Please choose a password, read and accept our agreements to proceed.

E-mail:

Password:

Confirm password:

Agreements: I've read and agree to the [Terms of Service](#) and [Privacy Policy](#)  
 Yes  No

Consent: Please email me updates regarding feature announcements, performance suggestions, feedback surveys and special offers  
 Yes  No

Submit

## 注册成功

List all instances ▾

 admin@vps6.win ▾

## Instances

+ Create New Instance

Name	Plan	Datacenter	Actions
------	------	------------	---------

You don't have any instances yet, do you want to [create one?](#)

## 选tw

## Create new instance

No credit card Please add a credit card if you want to subscribe to a paid plan

Missing company address Please fill in your address if you want to subscribe to a paid plan

Name: test

Plan: Little Lemur (Free)

Data center: asia-east1 (Taiwan)


Google Cloud Platform

Tags: test

Admins can manage tag access control.

Create New Instance

Plan



Little Lemur

See the plan page to learn about the different plans.

## Instances

+ Create New Instance

### test

Name	Host	Plan	Datacenter	Actions
test	mustang	Lemur	Google Compute Engine asia-east1 (Taiwan)	Edit RabbitMQ Manager

### test10

Name	Host	Plan	Datacenter	Actions
test10	mustang	Lemur	Google Compute Engine asia-east1 (Taiwan)	Edit RabbitMQ Manager

### test11

Name	Host	Plan	Datacenter	Actions
test11	mustang	Lemur	Google Compute Engine asia-east1 (Taiwan)	Edit RabbitMQ Manager

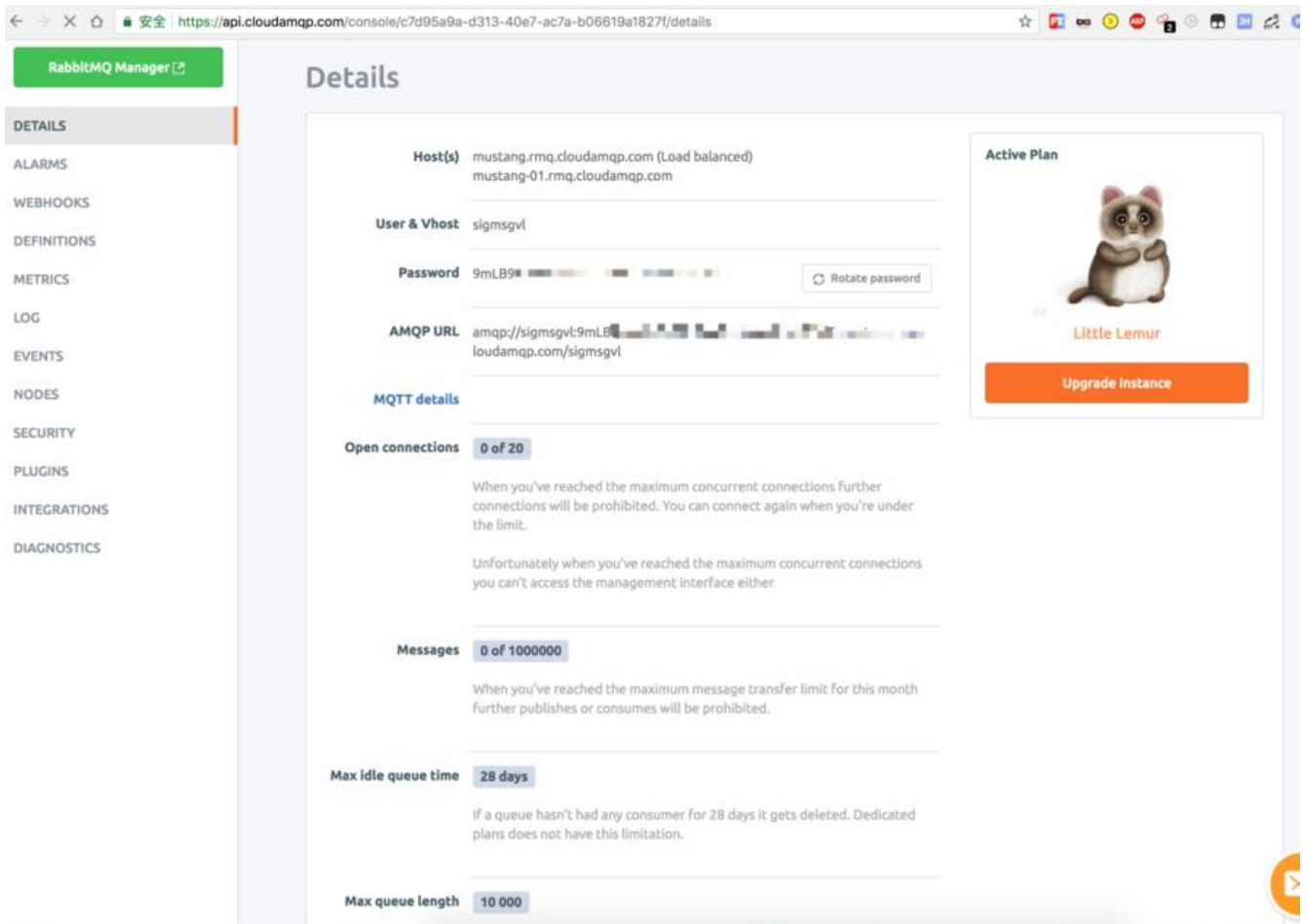
### test2

Name	Host	Plan	Datacenter	Actions
test2	mustang	Lemur	Google Compute Engine asia-east1 (Taiwan)	Edit RabbitMQ Manager

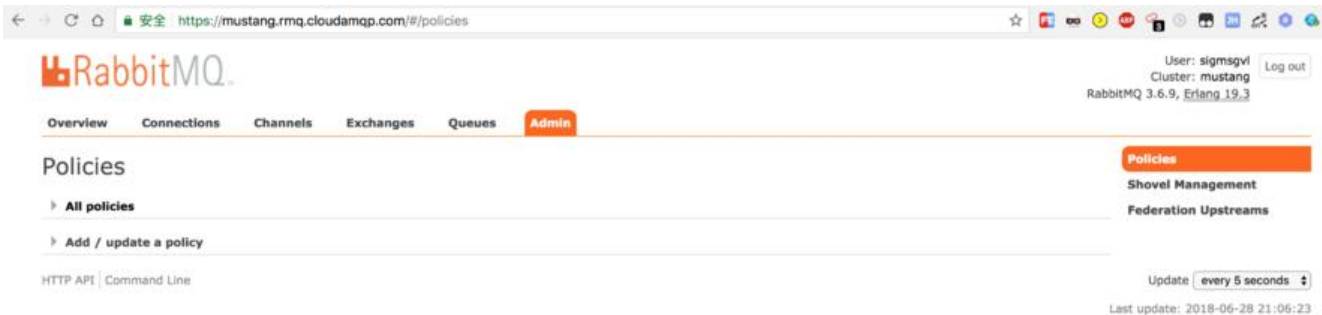
## 服务详情

免费的嘛，有限制

- Max 100 queues
- Max 10 000 queued messages
- Max idle queue time 28 days
- Max 20 connections
- Max queue length 10000
- Messages 1000000



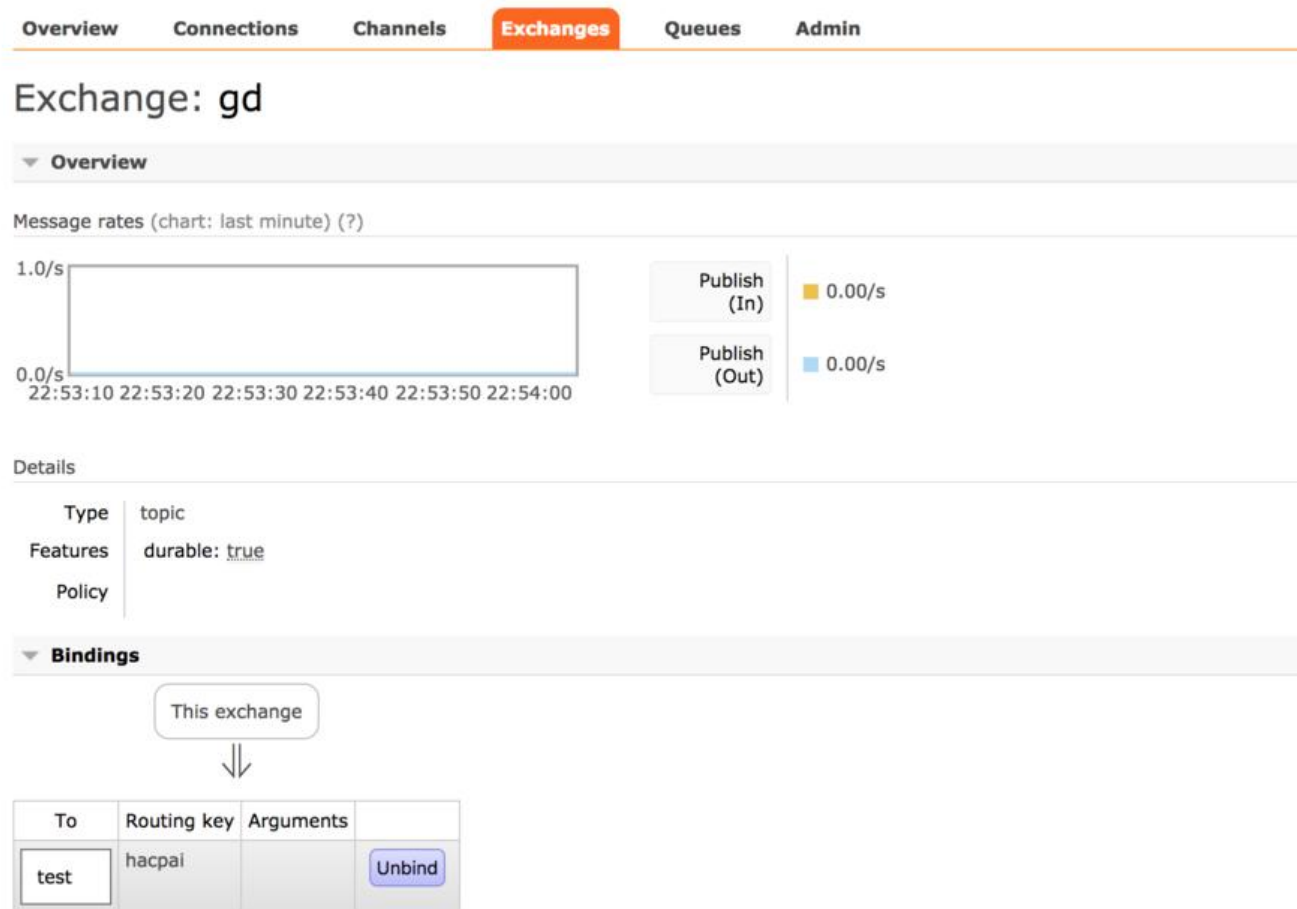
## rabbitmq管理平台



## 测试

安装: `pip install pika`

## 交换机,队列设置



## 生产者



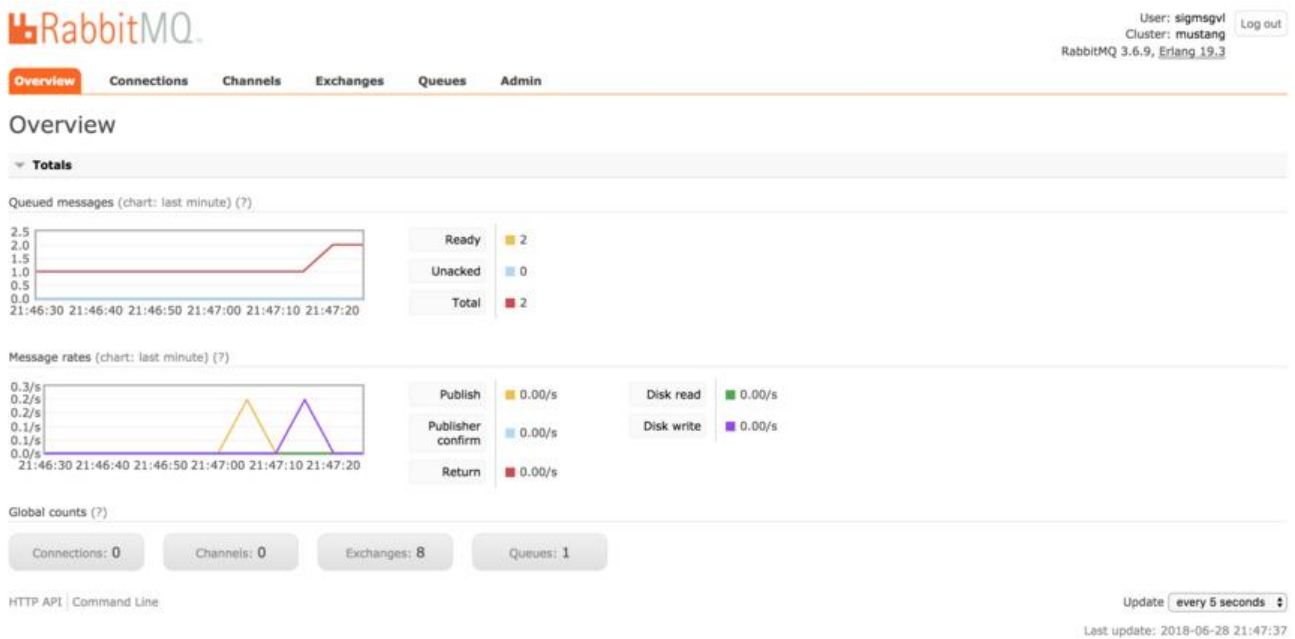
```

x sta 1 # -*- coding:utf-8 -*-
2
* RE 3 import pika
x you 4 import sys
5
x api 6 username = 'sigmsgvl' #指定远程rabbitmq的用户名密码
x api 7 pwd = '9mLB9oxmSJFo7K2vvC-qDnohDKek5fN'
x con 8 user_pwd = pika.PlainCredentials(username, pwd)
* inp 9 s_conn = pika.BlockingConnection(pika.ConnectionParameters('mustang.rmq.cloudamqp.com',5672,'sigmsgvl', credentials=user_pwd))#创建连接
* rab 10 chan = s_conn.channel() #在连接上创建一个频道
11
12 chan.queue_declare(queue='test',durable=True) #声明一个队列。生产者和消费者都要声明一个相同的队列，用来防止万一某一方挂了，另一方能正常运行
13 chan.basic_publish(exchange='gd', #交换机
14 routing_key='hacpai', #路由键。写明将消息发往哪个队列，本例是将消息发往队列hello
15 body='hello hacpai')#生产者要发送的消息
16 print("[生产者] send 'hello hacpai'")
17
18 s_conn.close()#当生产者发送完消息后，可选择关闭连接
19
20

```

[生产者] send 'hello hacpai  
[Finished in 1.4s]

## 面板信息



## 消息详情

安全 | <https://mustang.rmq.cloudamqp.com/#/queues/sigmsgvl/test>

Get (auto ack) 0.00/s

Details

Features	State	Total	Ready	Unacked	In memory	Persistent	Transient, Paged Out
Policy: HA	running	2	2	0	0	0	2
Consumers: 0	Consumer utilisation (?): 0%	Message body bytes (?): 25B	25B	0B	0B	0B	25B
		Process memory (?): 140kB					

Consumers

Bindings

Publish message

Get messages

Warning: getting messages from a queue is a destructive action. (?)

Requeue: Yes

Encoding: Auto string / base64 (?)

Messages: 1

Get Message(s)

Message 1

The server reported 1 messages remaining.

Exchange	gd
Routing Key	hacpai
Redelivered	0
Properties	
Payload	13 bytes: Hello hacpai
Encoding	string

## 消费者

```

1 # -*- coding:utf-8 -*-
2
3 import pika
4 import sys
5
6 username = 'sigmsgvl' #指定远程rabbitmq的用户名密码
7 pwd = '9mLB9oxmSJFo07K2vvC-qDnohDKek5fN'
8 user_pwd = pika.PlainCredentials(username, pwd)
9 connection = pika.BlockingConnection(pika.ConnectionParameters('mustang.rmq.cloudamqp.com',5672,'sigmsgvl', credentials=user_pwd))#创建连接
10 channel = connection.channel() #在连接上创建一个频道
11
12 # 声明消息队列，消息将在这个队列中进行传递。如果队列不存在，则创建
13 channel.queue_declare(queue='test',durable=True)
14
15 # 定义一个回调函数来处理，这边的回调函数就是将信息打印出来。
16 def callback(ch, method, properties, body):
17     print("[x] Received %r" % body)
18
19 # 告诉rabbitmq使用callback来接收信息
20 channel.basic_consume(callback,
21                       queue='test',
22                       no_ack=True)
23 # no_ack=True表示在回调函数中不需要发送确认标识
24
25 print('[*] Waiting for messages. To exit press CTRL+C')
26
27 # 开始接收信息，并进入阻塞状态，队列里有信息才会调用callback进行处理。按ctrl+c退出。
28 channel.start_consuming()
29

```

```

[*] Waiting for messages. To exit press CTRL+C
[x] Received 'Hello hacpai!'
[x] Received 'hello hacpai'

```

## 面板信息

## Overview

## Totals

Queued messages (chart: last minute) (?)



Ready	0
Unacked	0
Total	0

Message rates (chart: last minute) (?)



Publish	0.00/s	Deliver (auto ack)	0.00/s	Get (manual ack)	0.00/s
Publisher confirm	0.00/s	Consumer ack	0.00/s	Get (auto ack)	0.00/s
Deliver (manual ack)	0.00/s	Redelivered	0.00/s	Return	0.00/s

Disk read	0.00/s
Disk write	0.00/s

Global counts (?)

Connections: 1	Channels: 1	Exchanges: 8	Queues: 1
----------------	-------------	--------------	-----------

[HTTP API](#) | [Command Line](#)Update: every 5 seconds  
Last update: 2018-06-28 21:54:40

## 参考资料:

<https://www.cnblogs.com/panguoping/p/5720134.html><https://www.cloudamqp.com/docs/index.html>

## 代码

```
# -*- coding:utf-8 -*-
import pika
import sys

username = 'sigmsgvl' #指定远程rabbitmq的用户名密码
pwd = '9mLB9oxmSJFo07K2vvC-qDnohDKek5fN'
user_pwd = pika.PlainCredentials(username, pwd)
s_conn = pika.BlockingConnection(pika.ConnectionParameters('mustang.rmq.cloudamqp.com'
5672,'sigmsgvl', credentials=user_pwd))#创建连接
chan = s_conn.channel() #在连接上创建一个频道
chan.queue_declare(queue='test',durable=True) #声明一个队列，生产者和消费者都要声明一个
同的队列，用来防止万一某一方挂了，另一方能正常运行
chan.basic_publish(exchange='gd', #交换机
routing_key='hacpai',#路由键，写明将消息发往哪个队列，本例是将消息发往队列hello
body='hello hacpai')#生产者要发送的消息
print("[生产者] send 'hello hacpai")
s_conn.close()#当生产者发送完消息后，可选择关闭连接

# -*- coding:utf-8 -*-
import pika
import sys

username = 'sigmsgvl' #指定远程rabbitmq的用户名密码
pwd = '9mLB9oxmSJFo07K2vvC-qDnohDKek5fN'
```

```
user_pwd = pika.PlainCredentials(username, pwd)
connection = pika.BlockingConnection(pika.ConnectionParameters('mustang.rmq.cloudamqp.
om',5672,'sigmsgvl', credentials=user_pwd))#创建连接
channel = connection.channel() #在连接上创建一个频道
channel.queue_declare(queue='test',durable=True) # 声明消息队列，消息将在这个队列中进行
递。如果队列不存在，则创建
# 定义一个回调函数来处理，这边的回调函数就是将信息打印出来。
def callback(ch, method, properties, body):
    print(" [x] Received %r" % body)
# 告诉rabbitmq使用callback来接收信息
channel.basic_consume(callback,
                        queue='test',
                        no_ack=True)
# no_ack=True表示在回调函数中不需要发送确认标识
print('[*] Waiting for messages. To exit press CTRL+C')
# 开始接收信息，并进入阻塞状态，队列里有信息才会调用callback进行处理。按ctrl+c退出。
channel.start_consuming()
```