How to add a timeout to method start_consuming() on pika library

Asked 4 years, 3 months ago Modified 14 days ago Viewed 8k times

3

I have a `BlockingConnection`, and I follow the examples of pika documentation. But in all of them, the example of code to start consuming messages are:

```
connection = pika.BlockingConnection()
channel = connection.channel()
channel.basic_consume('test', on_message)
try:
    channel.start_consuming()
except KeyboardInterrupt:
    channel.stop_consuming()
connection.close()
```

(with more or less details).

I have to code many scripts, and I want to run one after another (for test/research purposes). But the above code require that I added ^C in each one.

I try to add some timeouts explained in the documentation, but I haven't luck. For example, if I find a parameter for set if client don't consuming any message in the last X seconds, then script finish. Is this posible in pika lib? or I have to change the approach?

python rabbitmq pika

Share Improve this question Follow

asked Jul 12, 2019 at 2:23

Tuxman

378 ● 4 ● 13

1 You want your code to automatically kill the consumer after a certain amount of time. Is that right? – bumblebee Jul 12, 2019 at 6:28

@bumblebee Ok, thats could be an option. But this "amount of time" should be after don't exists more message in the queue. For Example, in C++ client you can to set a timeout. – Tuxman Jul 12, 2019 at 11:31

3 Answers

Sorted by: Highest score (default)

5

Don't use `start_consuming` if you don't want your code to block. Either use `SelectConnection` or this method that uses `consume`. You can add a timeout to the parameters passed to `consume`.

Share Improve this answer Follow

edited Oct 12 at 19:37

Dharman ♦

31.2k ● 25 ● 87 ● 139

answered Jul 12, 2019 at 15:23

Luke Bakken

8,920 ● 2 ● 20 ● 35

Using the BlockingChannel.consume generator to consume messages

The `BlockingChannel.consume` method is a generator that will return a tuple of method, properties and body.

When you escape out of the loop, be sure to call consumer.cancel() to return any unprocessed messages.

Example of consuming messages and acknowledging them:

```
import pika

connection = pika.BlockingConnection()
channel = connection.channel()

# Get ten messages and break out
for method_frame, properties, body in channel.consume('test'):

    # Display the message parts
    print(method_frame)
    print(properties)
    print(body)

    # Acknowledge the message
    channel.basic_ack(method_frame.delivery_tag)

    # Escape out of the loop after 10 messages
    if method_frame.delivery_tag == 10:
        break

# Cancel the consumer and return any pending messages
requeued_messages = channel.cancel()
print('Requeued %i messages' % requeued_messages)

# Close the channel and the connection
channel.close()
connection.close()
```

If you have pending messages in the test queue, your output should look something like:

```
(pika)gmr-0x02:pika gmr$ python blocking_nack.py
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=1',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=2',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=3',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=4',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=5',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=6',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=7',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=8',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=9',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
<Basic.Deliver(['consumer_tag=ctag1.0', 'redelivered=True', 'routing_key=test', 'delivery_tag=10',
'exchange=test'])>
<BasicProperties(['delivery_mode=1', 'content_type=text/plain'])>
Hello World!
Requeued 1894 messages
```

该方法还可以指定最大等待时间inactivity_timeout参数，在队列为空时可以及时返回：
for method_frame, properties, body in channel.consume('test', inactivity_timeout=0.5)