

 NPException  
码龄4年 暂无认证

465  
原创

1万+  
周排名

2137  
总排名

160万+  
访问

 等级

1万+  
积分

362  
粉丝

471  
获赞

169  
评论

1396  
收藏













私信

关注

搜博主文章

Q

分类专栏

	java	97篇
	java8	2篇
	spring	12篇
	springBoot	41篇
	springCloud	56篇
	mybatis	7篇
▼		

最新评论

spring boot返回json数据和完美使用fastjs...

公众号：Java Pro：一键三连，给你磕头 ...

Logback详解

佚名留香: 

Mapper.xml详解

qq\_42920672: 请问能把你的User也展示一下吗

Java SimpleDateFormat 中英文时间格式...

taoqeyongwu: 谢谢大佬

springboot连接elasticsearch 报错failed t...

wph-01: 赞，版本问题，一直没找到！

最新文章

Centos7 系统根分区空间小,home空间大。怎么删除home分区 增加到分区			
使用stream流将list集合中某一BigDecimal字段求和			
Logback详解			
2021年	1篇	2020年	12篇
2019年	355篇	2018年	194篇

目录

Springboot2.x集成RabbitMQ实现消费者...

前言

yml配置

## 6.Springboot2.x集成RabbitMQ实现消费者限流，手动ack确认

原创

NPException

2020-02-13 13:50:36

 1377

 收藏 5

版权

分类专栏: # rabbitmq

## Springboot2.x集成RabbitMQ实现消费者限流，手动ack确认

### 前言

我们在实际项目中，可能在mq中积累了成千上万的消息，如果我们不进行限流，当我们打开消费者的时候一下子成千上万的消息一下子冲击过来，可能会造成服务器宕机，或者业务出现严重漏洞，所以我们需要进行消费者限流。首先我的springboot版本，springBootVersion = '2.2.1.RELEASE'。其他版本配置差别都不大。

注意：在版本2.0之前的版本中，只有一种MessageListenerContainer—SimpleMessageListenerContainer;在2.0之后有第二个容器——DirectMessageListenerContainer，配置如下：

```
1 listener:
2 #Container where the RabbitMQ consumer dispatches messages to an invoker thread
3 type: simple
4 simple:
5   acknowledge-mode: manual
6   prefetch: 1
7 #Container where the listener is invoked directly on the RabbitMQ consumer thread
8 type: direct
9 direct:
10   acknowledge-mode: manual
11   prefetch: 1
```

### SimpleMessageListenerContainer

默认情况下，侦听器容器将启动单个使用者，该使用者将从队列接收消息。

在检查上一节中的表时，您将看到许多控制并发性的属性。最简单的是concurrentConsumers，它只创建(固定的)并将发处理消息的使用者数量。

此外，还添加了一个新的属性maxConcurrentConsumers，容器将根据工作负载动态调整并发性。这与四个附加属性一起工作:continutiveactivetrigger、startConsumerMinInterval、continutiveidletrigger、stopConsumerMinInterval。

在默认设置下，增加消费者的算法工作如下：

如果尚未到达maxConcurrentConsumers，并且已有的使用者连续10个周期处于活动状态，并且自上一个使用者启动以来至少已经过了10秒，那么将启动一个新的使用者。如果使用者在txSize \*中接收到至少一条消息，则认为该使用者处于活动状态。

在默认设置下，减少消费者的算法工作如下：

如果有多个concurrentConsumers正在运行，并且某个consumer检测到10个连续超时(空闲)，并且上一个consumer至少在60秒之前停止，那么该consumer将停止。超时取决于receiveTimeout和txSize属性。如果使用者在txSize \*中没有接收到任何消息，则认为它是空闲的。因此，在默认超时(1秒)和txSize为4的情况下，在40秒的空闲时间(4个超时对应1个空闲检测)之后将考虑停止使用者。

### DirectMessageListenerContainer

使用DirectMessageListenerContainer，您需要确保ConnectionFactory配置了一个任务执行器，该执行器在使用该ConnectionFactory的所有侦听器容器中具有足够的线程来支持所需的并发性。默认连接池大小仅为5。

并发性基于配置的队列和consumersPerQueue。每个队列的每个使用者使用一个单独的通道，并发性由rabbit客户端库控制;默认情况下，它使用5个线程池;您可以配置taskExecutor来提供所需的最大并发性。

### 对比

SimpleMessageListenerContainer提供了以下特性，但DirectMessageListenerContainer不提供:

- txSize—使用SimpleMessageListenerContainer，您可以将其设置为控制事务中传递的消息数量和/或减少ack的数量，但这可能会导致失败后重复传递的数量增加。(与txSize和SimpleMessageListenerContainer一样，DirectMessageListenerContainer也有messagesPerAck，可以用来减少ack，但不能用于事务—每个消息都在单独的事务中交付和打包)。
- maxconcurrentconsumer和consumer伸缩间隔/触发器—DirectMessageListenerContainer中没有自动伸缩;但是，它允许您以编程方式更改consumersPerQueue属性，并相应地调整使用者。

然而，与SimpleMessageListenerContainer相比，DirectMessageListenerContainer有以下优点:

- 在运行时添加和删除队列更有效;使用SimpleMessageListenerContainer，整个使用者线程重新启动(所有使用者取消并重新创建);对于DirectMessageListenerContainer，不受影响的使用者不会被取消。避免了RabbitMQ客户机线程和使用者线程之间的上下文切换。
- 线程是跨使用者共享的，而不是为SimpleMessageListenerContainer中的每个使用者都有一个专用线程。但是，请参阅"线程和异步使用者"一节中有关连接工厂配置的重要说明。

### yml配置

```
1 spring:
2   application:
3     name: zoo-plus-rabbitmq
4   rabbitmq:
5     virtual-host: /
6     host: localhost
7     port: 5672
8     username: guest
9     password: guest
10    listener:
11      type: simple
12      simple:
13        acknowledge-mode: manual #采用手动应答
14        prefetch: 1 #限制每次发送一条数据。
```

新建一个队列：

```
1 /**
2  * 测试队列
3  */
4 @Bean
5 public Queue testQueue() {
6     return QueueBuilder.nonDurable("test-queue").build();
7 }
```

生产者：

```
1
2 /**
3  * 发送五条数据，测试消费端必须ack才发送第二条，消费者限流
4  */
5 @GetMapping("ack")
6 public Resp testAck() {
7     for (int i = 0; i < 5; i++) {
8         rabbitTemplate.convertAndSend("test-queue", "测试ack确认模式");
9     }
10    return Resp.success("ok", null);
11 }
```





```
1      @RabbitListener(queues = {"test-queue"})
2      public void testQueue(Message message, Channel channel) throws IOException {
3          log.info("test-queue消费: " + new String(message.getBody()));
4
5          /*
6              listener:
7              type: simple
8              simple:
9              #采用手动应答
10              acknowledge-mode: manual
11              prefetch: 1 #限制每次发送一条数据。
12          */
13      //      采用手动ack, 一条条的消费
14      channel.basicAck(message.getMessageProperties().getDeliveryTag(), false);
15
16      /*
17      * 还可以nack
18      * 第三个参数是否重回队列
19      */
20      //      channel.basicNack(message.getMessageProperties().getDeliveryTag(),false,t
21      }
```

由上面测试，如果我们没有配置，一下子五条信息就全部拥挤上来了，如果我们加了配置，不在消费端进行ack的话，消息就会unacked，下次重启服务也会再一次发送这条消息，直到消费端ack。

源码地址：https://gitee.com/zoo-plus/springboot-learn/tree/2.x/springboot-middlewre/rabbitmq

SpringBoot Rabbitmq 消息发送手动确认demo

SpringBoot Rabbitmq 消息发送手动确认demo,通过实现RabbitTemplate.ConfirmCallback 的confirm方法来手动确认

10-30

SpringBoot集成RabbitMq手动确认消息ACK（亲测）

SpringBoot集成RabbitMq手动确认消息ACK（亲测） Linux部署环境 采用Docker快速部署rabbitMq环境 docker安装 安装yu...

Geekelove的博文 5300

 优质评论可以帮助作者获得更高权重

抢沙发

 评论

相关推荐

- springboot整合rabbitmq 消费者Consumer 手动进行ack确...

ack指Acknowledge,确认。表示消费端收到消息后的确认方式。有三种确认方式:自动确认:acknowledge="none"手动确认:a...

4-12
- springboot2.x集成RabbitMQ实现消息发送确认 与 消息接收确认(ACK)

前言首先看回调机制:消息不管是否投递到交换机都进行ConfirmCallback回调,投递成功ack=true,否则为false 交换机匹配到...

4-5
- SpringBoot+RabbitMQ发送确认和消费手动确认机制

配置RabbitMQ # 发送确认 spring.rabbitmq.publisher-confirms=true # 发送回调 spring.rabbitmq.publisher-returns=true # 消...

yuyeqianhen的博文 8724
- springboot + rabbitmq 消费者消息确认 （Ack）

springboot + rabbitmq 消费者消息确认 （Ack） 开启消息确认 spring.rabbitmq.listener.direct.acknowledge-mode>manual s...

weixin\_43866295的博文 3748
- SpringBoot整合RabbitMq生产者和消费者消息确认机制(ack)

SpringBoot整合RabbitMq生产者和消费者消息确认机制(ack)首先引入maven依赖 <dependency> <groupId>org.springfram...

4-14
- springboot整合RabbitMQ 消费者消息接收确认

一，消息接收确认 1.ACK机制：消息确认机制 1.作用： 确认消息是否被消费者消费，消息通过ACK机制确认是否被正确接...

kevin的博文 1万+
- SpringBoot+RabbitMQ实现手动Consumer Ack

@[TOC](目录) 一、 Consumer Ack的三种方式 (1)、自动确认： acknowledge = "none",这是默认的方式,如果不配置的话,默...

LoveLacie的博文 1530
- springboot集成rabbitmq并手动注册容器实现单个queue的ack模式

rabbitmq的基础内容在之前已经介绍过，若有疑问，可参考我的之前的博文RabbitMQ基础介绍 接下来本篇博文中出现的代...

hhsway的博文 2127
- springboot2.3.1整合RabbitMQ多种工作模式 发送确认，手动应答

springboot2.3.1整合RabbitMQ多种工作模式 发送确认，手动应答

胖虎儿的博文 2142
- 手把手教你SpringBoot+RabbitMQ实现手动Consumer Ack

你知道的越多，不知道的就越多，业余的像一棵小草！ 你来，我们一起精进！你不来，我和我的竞争对手一起精进！ 编辑： ...

xmt1139057136的专栏 221
- springBoot-rabbit MQ-设置手动确认ACK-Channel

自动确认消费，不管 try 中有没有异常，消息管理界面上队列里的消息都被消费了，没有ready和unacked状态栏； 手动确...

fwk19840301的博文 4983
- Spring Boot RabbitMq 并发与限流

概述 电商中秒杀请求，属于瞬间大流量，同一时刻会有大量的请求涌入到系统中，可能导致系统挂掉。应付这种瞬间大流...

Sam哥哥聊技术 6001
- SpringBoot集成RabbitMQ(注解+手动确认)

1.pom文件 <dependency> <groupId>org.springframework.boot</groupId> <artifactId>spring-boot-starter-amqp</artifactId>...

795
- Springboot2.x集成Rabbitmq实现消费者限流，手动ack确认

前言 我们在实际项目中，可能在mq中积累了成千上万的消息，如果我们不进行限流，当我们打开消费者是时候一下子成千...

霓虹深处 1521
- Springboot整合Rabbitmq实现消费者手动确认（转载）

https://www.cnblogs.com/haixiang/p/10959551.html(侵必删)

xiaoBaiaaaaaa的博客 69
- springboot + rabbitMQ 消费端限流限流

新建 RabbitMQProducer项目后，添加依赖 <dependency> <groupId>org.springframework.boot</groupId> <artifactId>spri...

qq\_41784817的博文 619
- RabbitMq学习——Springboot整合rabbitmq之手动消息确认(ACK)

一、前言 前几天我研究了关于springboot整合简单消息队列，实现springboot推送消息至队列中，消费者成功消费。同时也...

qq\_38322527的博文 4295
- SpringBoot与ActiveMQ整合实现手动ACK

看这篇文章之前，我相信大家已经有过ActiveMQ的基本知识，已经知道JmsTemplate、MessageListenerContainer、Conne...

二月初二 4139
- springboot-rabbitmq ack的问题？

如何手动ack - 网上的方式1： 实现channelAwareMessageListener public class RabbitMsgReceiver implements ChannelA...

java\_seeking的博文 868
- SpringBoot集成RabbitMQ消息队列搭建与ACK消息确认入门

1.Windows下安装RabbitMQ的步骤详解+图解（erlang+RabbitMQ） 2.SpringBoot集成RabbitMQ参考文章1.RabbitMQ介绍...

屎壳郎情调-成长日记 3万+
- springboot+activemq手动处理

1. springboot框架中activemq手动连接处理 2. activemq事务性分析 记录一个最近使用activemq的使用总结，场景是activem...

loney\_wolf的博文 981

©2020 CSDN 皮肤主题: 技术黑板 设计师:CSDN官方博客 返回首页

关于我们 招贤纳士 广告服务 开发助手 400-660-0108 kefu@csdn.net 在线客服 工作时间 8:30-22:00  
公安备案号11010502030143 京ICP备19004658号 京网文〔2020〕1039-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心 网络110报警服务 中国互联网举报中心 家长监护 Chrome商店下载 ©1999-2021北京创新乐知网络技术有限公司 版权与免责声明 版权申诉 出版物许可证 营业执照



举报