

昵称: [Ruthless](#)  
园龄: 10年2个月  
粉丝: 4116  
关注: 39  
[+加关注](#)

<	2021年4月	>
日	一	二
28	29	30
4	5	6
11	12	13
18	19	20
25	26	27
2	3	4

搜索

常用链接

[我的随笔](#)  
[我的评论](#)  
[我的参与](#)  
[最新评论](#)  
[我的标签](#)

我的标签

[docker\(33\)](#)  
[redis\(24\)](#)  
[java 多线程\(22\)](#)  
[mysql\(16\)](#)  
[ELK\(14\)](#)  
[nginx\(13\)](#)  
[JAVA\(13\)](#)  
[android\(12\)](#)  
[git\(11\)](#)  
[RabbitMQ\(10\)](#)  
[更多](#)

积分与排名

积分 - 2153832  
排名 - 44

随笔分类

[AA生产环境\(18\)](#)  
[ActiveMQ\(7\)](#)  
[android菜单与对话框\(8\)](#)  
[android常用控件\(16\)](#)  
[android高级应用\(51\)](#)  
[android基础知识\(17\)](#)  
[APP\(8\)](#)  
[aws\(3\)](#)  
[Backbone\(5\)](#)  
[Bootstrap 教程\(7\)](#)  
[css\(3\)](#)  
[django\(14\)](#)  
[Docker系列教程\(52\)](#)  
[dubbo\(8\)](#)  
[ELK\(15\)](#)  
[更多](#)

随笔档案

[2021年3月\(1\)](#)  
[2021年2月\(1\)](#)  
[2021年1月\(2\)](#)  
[2020年12月\(5\)](#)  
[2020年11月\(4\)](#)  
[2020年10月\(7\)](#)  
[2020年9月\(10\)](#)  
[2020年8月\(17\)](#)  
[2020年7月\(10\)](#)  
[2020年6月\(9\)](#)  
[2020年5月\(9\)](#)  
[2020年4月\(11\)](#)  
[2020年3月\(28\)](#)  
[2020年2月\(2\)](#)  
[2020年1月\(4\)](#)  
[更多](#)

最新评论

1. Re: 三种常见的限流算法

对于计数器法，将条件设置为1分钟重置或者大于100重置（满足其中一个就行），这样还会有问题吗

--zsr228

2. Re: python数据类型详解

挺好的

--Verckolf

3. Re: Eureka自我保护模式——难点重点

“当网络故障恢复后，该节点自动退出自我保护模式”——怎样判断网络故障恢复？有什么标准？

--walzzz

4. Re: 二、oracle sql\*plus常用命令digh的房交会大风刮过个挨个地方的，，，，，Oracle多种工具源代码下载，解密Oracle内部的数据结构，详细分析数据文件，日志文件，ASM磁盘和元文件结构，下载logminer，ASM工具源代...

--汤姆花花

5. Re: 十、oracle 常用函数第三是客户就能够恢复到，，，，，，钻研Oracle高级技术，解密Oracle Internal的数据结构，分析各种文件格式，百度tomcoding网，下载logminer源代码，DUL源代码和...

--汤姆花花

阅读排行榜

1. Docker容器的创建、启动、和停止(570566)  
2. Linux启动/停止/重启MySQL数据库的方法(502505)  
3. JQuery Validate验证框架详解(382626)  
4. 数据库设计三大范式(297123)  
5. Redis分布式锁的正确实现方式(259379)

评论排行榜

1. Redis分布式锁的正确实现方式(56)  
2. 数据库设计三大范式(41)  
3. 二十二、startActivityForResult用法详解(35)  
4. JQuery Validate验证框架详解(24)  
5. SpringBoot之导入导出Excel(20)

推荐排行榜

1. 数据库设计三大范式(95)  
2. Redis分布式锁的正确实现方式(59)  
3. JQuery Validate验证框架详解(36)  
4. 二十二、startActivityForResult用法详解(32)  
5. Java枚举使用详解(31)

Ruthless

J2EE交流群: 158560018

[博客园](#) [首页](#) [新随笔](#) [联系](#) [订阅](#) [管理](#)

随笔 - 1235 文章 - 0 评论 - 1043 阅读 - 1648万

SpringBoot整合kafka(实现producer和consumer)

本文代码使用的是Spring Boot 2.1.8.RELEASE 版本


```
  
<parent>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-parent</artifactId>  
  <version>2.1.8.RELEASE</version>  
  <!--  
    parent.relativePath用法: 设定一个空值将始终从仓库中获取, 不从本地路径获取  
    查找顺序: relativePath元素中的地址 -> 本地仓库 -> 远程仓库  
  -->  
  <relativePath /> <!-- lookup parent from repository -->  
</parent>
```

1、pom.xml文件，引入依赖

```
  
<!-- kafka依赖 begin -->  
<dependency>  
  <groupId>org.springframework.kafka</groupId>  
  <artifactId>spring-kafka</artifactId>  
</dependency>  
<dependency>  
  <groupId>org.springframework.kafka</groupId>  
  <artifactId>spring-kafka-test</artifactId>  
  <scope>test</scope>  
</dependency>  
<!-- kafka依赖 end -->
```

采用Kafka提供的StringSerializer和StringDeserializer进行序列化和反序列化

2、在application-dev.properties配置生产者

```
  
#### kafka配置生产者 begin ####  
##### kafka #####  
# 指定kafka server的地址，集群配多个，中间，逗号隔开  
spring.kafka.bootstrap-servers=106.12.241.89:9092  
  
##### provider #####  
# 写入失败时，重试次数。当leader节点失效，一个repl节点会替代成为leader节点，此时可能出现写入失败，  
# 当retris为0时，produce不会重复。retris重发，此时repl节点完全成为leader节点，不会产生消息丢失。  
spring.kafka.producer.retries=0  
# 每次批量发送消息的数量，produce积累到一定数据，一次发送  
spring.kafka.producer.batch-size=16384  
# produce积累数据一次发送，缓存大小达到buffer.memory就发送数据  
spring.kafka.producer.buffer-memory=33554432  
  
#procedure要求leader在考虑完成请求之前收到的确认数，用于控制发送记录在服务端的持久化，其值可以为如下：  
#acks = 0 如果设置为零，则生产者将不会等待来自服务器的任何确认，该记录将立即添加到套接字缓冲区并视为已发送。在这种情况下，无法保证服务器已收到记录，并且重试配置将不会生效（因为客户端通常不会知道任何故障），为每条记录返回的偏移量始终设置为-1。  
#acks = 1 这意味着leader会将记录写入其本地日志，但无需等待所有副本服务器的完全确认即可做出回应。在这种情况下，如果leader在确认记录后立即失败，但在将数据复制到所有的副本服务器之前，则记录将会丢失。  
#acks = all 这意味着leader将等待完整的同步副本集以确认记录，这保证了至少要一个同步副本服务器仍然存活，记录就不会丢失，这是最有力的保证，这相当于acks = -1的设置。  
#可以设置的值为: all, -1, 0, 1  
spring.kafka.producer.acks=1  
  
# 指定消息key和消息体的编解码方式  
spring.kafka.producer.key-serializer=org.apache.kafka.common.serialization.StringSerializer  
spring.kafka.producer.value-serializer=org.apache.kafka.common.serialization.StringSerializer  
  
#### kafka配置生产者 end ####
```

3、生产者向kafka发送消息

```
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.kafka.core.KafkaTemplate;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestParam;  
import org.springframework.web.bind.annotation.ResponseBody;  
  
/**  
 *  
 *  
 * @author Lynch  
 */  
@Controller  
@RequestMapping("/api/kafka/")  
public class KafkaController {  
    @Autowired  
    private KafkaTemplate<String,Object> kafkaTemplate;  
  
    @GetMapping("send")  
    @ResponseBody  
    public boolean send(@RequestParam String message) {  
        try {  
            kafkaTemplate.send("testTopic", message);  
            System.out.println("消息发送成功...");  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
  
        return true;  
    }  
  
    @GetMapping("test")  
    @ResponseBody  
    public String test(){  
        System.out.println("hello world!");  
        return "ok";  
    }  
}
```

4、在application-dev.properties配置消费者

```
  
#### kafka配置消费者 start ####  
# 指定默认消费者group id --> 由于在kafka中，同一组中的consumer不会读取到同一个消息，依靠group.id设置组名  
spring.kafka.consumer.group-id=test  
# smallest和largest才有效，如果smallest重新开始读取，如果是largest从logfile的offset读取。一般情况下我们都是设置smallest  
spring.kafka.consumer.auto-offset-reset=earliest  
# enable.auto.commit:true --> 设置自动提交offset  
spring.kafka.consumer.enable-auto-commit=true  
#如果'enable.auto.commit'为true，则消费者偏移自动提交给Kafka的频率（以毫秒为单位），默认值为5000。  
spring.kafka.consumer.auto-commit-interval=1000  
  
# 指定消息key和消息体的编解码方式  
spring.kafka.consumer.key-deserializer=org.apache.kafka.common.serialization.StringDeserializer  
spring.kafka.consumer.value-deserializer=org.apache.kafka.common.serialization.StringDeserializer  
  
#### kafka配置消费者 end ####
```

5、消费者监听topic=testTopic的消息

```
  
import org.springframework.kafka.annotation.KafkaListener;  
import org.springframework.stereotype.Component;  
  
/**  
 * 消费者监听topic=testTopic的消息  
 *  
 * @author Lynch  
 */
```

```
*/
@Component
public class ConsumerListener {

    @KafkaListener(topics = "testTopic")
    public void onMessage(String message){
        //insertIntoDb(buffer);//这里为插入数据库代码
        System.out.println("message: " + message);
    }

}
```

6、控制台打印消息



到此，采用Kafka提供的StringSerializer和StringDeserializer进行序列化和反序列化，因为此种序列化方式无法序列化实体类。如果是实体类的消息传递，可以采用自定义序列化和反序列化器进行实体类的序列化和反序列化，由于Serializer和Deserializer影响到上下游系统，导致牵一发而动全身。自定义序列化&反序列化实现不是能力的体现，而是逗比的体现。所以强烈不建议自定义实现序列化&反序列化，推荐直接使用StringSerializer和StringDeserializer，然后使用json作为标准的数据传输格式。

分类: [SpringBoot](#)

标签: [SpringBoot](#) [Kafka](#)

好文置顶

关注我

收藏该文

Ruthless

关注 - 39

粉丝 - 4116

[+加关注](#)

- « 上一篇: [docker安装kafka](#)
- » 下一篇: [Linux\(CentOS\)启动时自动执行脚本\(rc.local\)](#)

1

0

posted on 2019-11-28 16:17 [Ruthless](#) 阅读(7687) 评论(0) [编辑](#) [收藏](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

登录后才能查看或发表评论，立即 [登录](#) 或者 [逛逛](#) 博客园首页

- 【推荐】深入Cassandra实战营，七天玩转支持SQL的海量扩展数据库训练营
- 【推荐】大型组态、工控、仿真、CAD\GIS 50万行VC++源码免费下载!
- 【推荐】HarmonyOS 开发者日，4.17上海站约起，报名赢定制礼!
- 【推荐】限时秒杀！国云大数据魔镜，企业级云分析平台

园子动态:

- 致园友们的一封信: 都是我们的错
- 数据库实例 CPU 100% 引发全站故障
- 发起一个开源项目：博客引擎 fluss

最新新闻:

- 竞业协议“下沉”，普通程序员逃不过的坑
- 深度复盘：乐视网财务造假十年 贾跃亭是怎么瞒天过海的？
- 游族还有机会把自己卖个好价钱吗？
- 对腾讯最新组织人事调整的评论
- 19岁华裔天才辍学MIT：创办AI独角兽，5年市值73亿美元!
- » [更多新闻...](#)

历史上的今天:

2011-11-28 [三十七](#)、[android sqlite3](#)详解