Kafka实战

lizhitao1

目錄

My Awesome Book

This file file serves as your book's preface, a great place to describe your book's content and ideas.

```
package com.meituan.mq.demo.producer;

import com.meituan.mafka.client.MafkaClient;
import com.meituan.mafka.client.producer.IProducerProcessor;

import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.atomic.AtomicInteger;
import java.util.concurrent.atomic.AtomicLong;

/**
 * Created with IntelliJ IDEA.
 * User: lizhitao
 * Date: 15-3-10
 * Time: 下午4:58
 * To change this template use File | Settings | File Templates.
 */
public class MultiProducerMain {
//    static int count;
    static AtomicLong count = new AtomicLong();
    static volatile long sendRate = 0;
    static long start_time;
    static String sendStr = null;

    public static void main(String[] args) throws Exception {
//        IProducerProcessor producerProcessor2 = MafkaClient.buildMultiProducerFactory("
        IProducerProcessor producerProcessor2 = MafkaClient.buildProduceFactory("test-liz

        ProducerCountThread producerCountThread = new ProducerCountThread();
        producerCountThread.start();

        List<String> messages  = new ArrayList<String>();
        sendStr = "send msg xiaotao-lizherui-需要依赖Hibernate core 3.6.7.Final来编辑项目的源
        start_time = System.currentTimeMillis();
        for( int i = 1000; i < 40000000; i++ ) {
//            messages.add(sendStr + i);
//            if ( i % 20 == 0 ) {
//                producerProcessor2.sendMessages(messages);
                producerProcessor2.sendMessage(sendStr);

//                messages.clear();
//            }

            count.incrementAndGet();
//            sendRate++;
        }
    }


    static class ProducerCountThread extends Thread {

        public void run() {
            while (true) {
                try {
                    sleep(1000);
                    long end_time = System.currentTimeMillis();
                    long ellapsed =  end_time - start_time;
//                    System.out.println("count.get():" + count.get() + "  ellapsed:" + e
                        double totalInSec = (sendStr.getBytes().length + 4 ) * count.get() *
                        System.out.println(String.format("producer speed/sec:%d records/sec,
                                totalInSec / (1024.0 * 1024.0)
                        ,ellapsed / 1000));


                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }

            }
        }


        public synchronized static void countProducer() {
            sendRate++;
        }
    }
```