粒子物理后传之:利用Kafka api开发数据信息队列

kafka是一个分布式的、分区的、多复本的日志提交服务。

> 现在生产者一端是加速器内粒子束的探测数据，现在服务器需要一个消息队列来管理数据并发布给其他客户端(有可能有很多个，他们每一个可能各取所需)。这样的功能逻辑下我们可以沿用上一篇博客创建的Topic主题"EnengyInfo"来发布粒子能量数据。

使用maven构建java项目

我们需要使用maven构建一个叫Acc的项目，并且需要添加kafka依赖到pom.xml然后编译构建项目来从云端下载依赖包。 (假如下载依赖包速度慢可以自行下载kafka的jar添加到maven本地库路径)

```
[root@master ]# mvn archetype:generate -DgroupId=com.hanss.acc -DartifactId=
[root@master ]# vi pom.xml
<dependency>
   <groupId>org.apache.kafka</groupId>
   <artifactId>kafka_2.11</artifactId>
   <version>0.8.2.1</version>
</dependency>
[root@master ]# mvn package
```

利用kafka api编写生产/消费模型

我们采用本地模式，代码拆分为配置代码、生产者代码、消费者代码、主函数入口。

ConfigureAPI.java

```
package com.hanss.acc;

/**
 * @Date Nov 20, 2017
 *
 * @Author hanss401
 *
 * @Note Set param path
 */
public class ConfigureAPI {

    public interface KafkaProperties {
        public final static String ZK = "127.0.0.1:2181,127.0.0.1:2181,127.(
        public final static String GROUP_ID = "test_group1";
        public final static String TOPIC = "testnight";
        public final static String BROKER_LIST = "127.0.0.1:9092,127.0.0.1:
        public final static int BUFFER_SIZE = 64 * 1024;
        public final static int TIMEOUT = 20000;
        public final static int INTERVAL = 10000;
    }

}
```

JProducer.java

```java
package com.hanss.acc;

import java.util.Properties;

import kafka.javaapi.producer.Producer;
import kafka.producer.KeyedMessage;
import kafka.producer.ProducerConfig;

/**
 * @Date Nov 20, 2017
 *
 * @Author hanss401
 *
 * @Note Kafka JProducer
 */
public class JProducer extends Thread {

    private Producer<Integer, String> producer;
    private String topic;
    private Properties props = new Properties();
    private final int SLEEP = 1000 * 3;
    private double num [] = {1217.001,1733.223,569.331,667.824,1217.001,173

    public JProducer(String topic) {
        props.put("serializer.class", "kafka.serializer.StringEncoder");
        props.put("metadata.broker.list", "127.0.0.1:9092");
        producer = new Producer<Integer, String>(new ProducerConfig(props))
        this.topic = topic;
    }

    @Override
    public void run() {
        int offsetNo = 1;
        while (true) {
            String msg = new String("新增Alpha粒子束:" + num[offsetNo]+" Mev'
            System.out.println("发送了消息->[" + msg + "]");
            System.out.println("消息主题: 粒子信息收集");
            producer.send(new KeyedMessage<Integer, String>(topic, msg));
            offsetNo++;
            if(offsetNo==12)break;
            try {
                sleep(SLEEP);
            } catch (Exception ex) {
                ex.printStackTrace();
            }
        }
    }

}
```

JConsumer.java

```java
package com.hanss.acc;

import java.util.HashMap;
```

```java
import java.util.List;
import java.util.Map;
import java.util.Properties;

import com.hanss.acc.ConfigureAPI.KafkaProperties;
import kafka.consumer.Consumer;
import kafka.consumer.ConsumerConfig;
import kafka.consumer.ConsumerIterator;
import kafka.consumer.KafkaStream;
import kafka.javaapi.consumer.ConsumerConnector;

/**
 * @Date Nov 20, 2017
 *
 * @Author hanss401
 *
 * @Note Kafka Consumer
 */
public class JConsumer extends Thread {

    private ConsumerConnector consumer;
    private String topic;
    private final int SLEEP = 1000 * 3;

    public JConsumer(String topic) {
        consumer = Consumer.createJavaConsumerConnector(this.consumerConfig
        this.topic = topic;
    }

    private ConsumerConfig consumerConfig() {
        Properties props = new Properties();
        props.put("zookeeper.connect", KafkaProperties.ZK);
        props.put("group.id", KafkaProperties.GROUP_ID);
        props.put("zookeeper.session.timeout.ms", "40000");
        props.put("zookeeper.sync.time.ms", "200");
        props.put("auto.commit.interval.ms", "1000");
        return new ConsumerConfig(props);
    }

    @Override
    public void run() {
        Map<String, Integer> topicCountMap = new HashMap<String, Integer>()
        topicCountMap.put(topic, new Integer(1));
        Map<String, List<KafkaStream<byte[], byte[]>>> consumerMap = consum
        KafkaStream<byte[], byte[]> stream = consumerMap.get(topic).get(0);
        ConsumerIterator<byte[], byte[]> it = stream.iterator();
        while (it.hasNext()) {
            System.out.println("接收到消息->[" + new String(it.next().message
            try {
                sleep(SLEEP);
            } catch (Exception ex) {
                ex.printStackTrace();
            }
        }
    }

}
```

KafkaClient.java

```
package com.hanss.acc;

import com.hanss.acc.ConfigureAPI.KafkaProperties;
import com.hanss.acc.JConsumer;
import com.hanss.acc.JProducer;

/**
 * @Date Nov 20, 2017
 *
 * @Author hanss401
 *
 * @Note To run Kafka Code
 */
public class KafkaClient {

    public static void main(String[] args) {
        JProducer pro = new JProducer(KafkaProperties.TOPIC);
        pro.start();

        JConsumer con = new JConsumer(KafkaProperties.TOPIC);
        con.start();
    }

}
```

编译构建

运行前：确保zookeeper和kafka服务在运行。

```
[root@master ]# mvn package
... ...
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 4.218 s
[INFO] Finished at: 2017-11-20T14:22:58+08:00
[INFO] Final Memory: 19M/179M
[INFO] ------------------------------------------------------------------------


[root@master ]# mvn exec:java -Dexec.mainClass="com.hanss.acc.KafkaClient"
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building Acc 1.0-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- exec-maven-plugin:1.6.0:java (default-cli) @ Acc ---
log4j:WARN No appenders could be found for logger (kafka.utils.VerifiablePr
log4j:WARN Please initialize the log4j system properly.
发送了消息->[新增Alpha粒子束:1733.223 Mev]
消息主题：粒子信息收集
接收到消息->[新增Alpha粒子束:1733.223 Mev]
发送了消息->[新增Alpha粒子束:569.331 Mev]
消息主题：粒子信息收集
接收到消息->[新增Alpha粒子束:569.331 Mev]
发送了消息->[新增Alpha粒子束:667.824 Mev]
消息主题：粒子信息收集
接收到消息->[新增Alpha粒子束:667.824 Mev]
发送了消息->[新增Alpha粒子束:1217.001 Mev]
消息主题：粒子信息收集
接收到消息->[新增Alpha粒子束:1217.001 Mev]
```