

producer_consumer 範例

gradle

```
// kafka dependency
// https://mvnrepository.com/artifact/org.apache.kafka/kafka-clients
implementation("org.springframework.kafka:spring-kafka")
// https://mvnrepository.com/artifact/org.slf4j/slf4j-api
implementation("org.slf4j:slf4j-api:2.0.5")
// https://mvnrepository.com/artifact/org.slf4j/slf4j-simple
implementation("org.slf4j:slf4j-simple:2.0.5")
testImplementation("org.junit.jupiter:junit-jupiter-api:5.9.2")
testRuntimeOnly("org.junit.jupiter:junit-jupiter-engine")
```

properties

```
kafka.listen.start:true
spring.application.name=kafka_demo
spring.kafka.bootstrap-servers=10.20.30.40:9092
spring.kafka.producer.key-serializer=org.apache.kafka.common.serialization.StringSerializer
spring.kafka.producer.value-serializer=org.apache.kafka.common.serialization.StringSerializer
spring.kafka.producer.properties.retries=10
spring.kafka.consumer.key-deserializer=org.apache.kafka.common.serialization.StringDeserializer
spring.kafka.consumer.value-deserializer=org.apache.kafka.common.serialization.StringDeserializer
spring.kafka.consumer.group-id=group1
```

producer

```
@Controller
@RequestMapping("/api")
public class ProducerController {

    @Autowired
    private KafkaTemplate<String, String> kafkaTemplate;

    @GetMapping("/kafka")
    public ResponseEntity<String> send() throws InterruptedException {
        for (int i = 0; i < 100; i++) {
            kafkaTemplate.send("first", "data"+i, "data"+i);
//            Thread.sleep(1000);
        }
        return ResponseEntity.ok("{\"success:true}\"");
    }

}
```

consumer

```
package com.momo.appteam.liveapi.listener;

import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.springframework.kafka.annotation.KafkaListener;
import org.springframework.kafka.annotation.PartitionOffset;
import org.springframework.kafka.annotation.TopicPartition;
import org.springframework.stereotype.Component;

@Component
public class LiveStreamKafkaListener {

    @KafkaListener(topics = "first", autoStartup = "${kafka.listen.start:false}")
    public void consumeLiveStream(ConsumerRecord record)
    { // 參閱: 收到的 value
//        System.out.println("收到的信息: " + record.toString());
        Object topic = record.topic();
```

```

Object key = record.key();
Object value = record.value();
System.out.println("收到的信息: 【topic】 " + topic + " 【key】 " + key + " 【value】 " + value );
//ConsumerRecord(
// topic = momoLiveStream, partition = 0,
// leaderEpoch = 0, offset = 760,
// CreateTime = 1697695508640,
// serialized key size = 6,
// serialized value size = 9,
// headers = RecordHeaders(headers = [], isReadOnly = false),
// key = liveld, value = afasffasf
// )
}
@KafkaListener(
    groupId = "default",
    autoStartup = "${kafka.listen.start:false}" ,
    topicPartitions = {
        @TopicPartition(
            topic = "first",

            partitionOffsets = {
                @PartitionOffset(partition = "0",initialOffset = "0")
            }
        )
    }
)
})
public void consumeAllLiveStream(ConsumerRecord record){
    Object topic = record.topic();
    Object key = record.key();
    Object value = record.value();
    System.out.println("收到的信息: 【topic】 " + topic + " 【key】 " + key + " 【value】 " + value );
}
}

```

🔄修訂版本 #2

★由 treeman 建立於 23 🎱🎱🎱 2023 13:36:05

✍由 treeman 更新於 26 🎱@🎱🎱 2024 18:15:31