Práctica Kafka

Primero deberemos crear el **topic:**

root@debian:/home/kafka/kafka_2.11-2.4.0/bin# ./kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic practica

Created topic practica.

root@debian:/home/kafka/kafka_2.11-2.4.0/bin# ./kafka-topics.sh --list --zookeeper localhost:2181

__consumer_offsets
practica

Luego haremos que el **procuder** empiece a enviar el archivo:

root@debian:/home/kafka/kafka_2.11-2.4.0/bin# cat ../personal.json | ./kafka-console-producer.sh --broker-list localhost:9092 --topic practica > /dev/null

* en mi caso el archivo estaba un nivel por encima de ahí que haya puesto ../personal.json, pero deberia ser la ruta a dicho archivo.

Por último pondremos el **consumer** a ‘escuchar’:

root@debian:/home/kafka/kafka_2.11-2.4.0/bin# ./kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic practica –from-beginning

Para la segunda parte de la práctica crearemos un **consumer** en scala

```
import java.time.Duration
import java.time.temporal.ChronoUnit
import java.util.Properties

import org.apache.kafka.clients.consumer.KafkaConsumer

import scala.collection.JavaConverters._

object Consumer {

 def main(args: Array[String]): Unit = {

  // Aqui definiremos la configuracion
  val props: Properties = new Properties()
  props.put("group.id", "test")
  props.put("bootstrap.servers", "localhost:9092")
  props.put("key.deserializer", "org.apache.kafka.common.serialization.StringDeserializer")
  props.put("value.deserializer", "org.apache.kafka.common.serialization.StringDeserializer")
  // Declaramos el consumidor de kafca con la configuracion qu hemos definido
  val consumidor = new KafkaConsumer[String, String](props)
  // Declaramos el topic al que tiene que estar atento el consumido
  val topic = List("practica")

  try {
    consumidor.subscribe(topic.asJava)
    while (true) {
      val msg = consumidor.poll(Duration.of(1, ChronoUnit.SECONDS))
      // Aqui filtramos los menajes para eliminar los nombres elegidos
      msg.asScala
        .filter(it => !it.value().contains("Giavani") && !it.value().contains("Noell"))
        .foreach(m => println(m.value()))
    }
  } catch {
    case e: Exception => e.printStackTrace()
  } finally {
    consumidor.close()
  }
 }
}
```

Éste filtra los nombres de *“Giavani”* y *“Noell”*, y veremos el siguiente resultado:


```
{
  "id": 1,
  "first_name": "Jeanette",
  "last_name": "Penddreth",
  "email": "jpenddreth0@census.gov",
  "gender": "Female",
  "ip_address": "26.58.193.2"
}, {
  "id": 2,
  "last_name": "Frediani",
  "email": "gfredianil@senate.gov",
  "gender": "Male",
  "ip_address": "229.179.4.212"
}, {
  "id": 3,
  "last_name": "Bea",
  "email": "nbea2@imageshack.us",
  "gender": "Female",
  "ip_address": "180.66.162.255"
}, {
  "id": 4,
  "first_name": "Willard",
  "last_name": "Valek",
  "email": "wvalek3@vk.com",
  "gender": "Male",
  "ip_address": "67.76.188.26"
}
```