Kafka Training

Preconditions

```
$ sudo jps
2183 Jps
1752 SupportedKafka
1832 SchemaRegistryMain
1775 KafkaRestMain
1487 QuorumPeerMain
```

Java Process Name	Service Name	Platform
QuorumPeerMain	zookeeper	hadoop
SupportedKafka/Kafka	kafka-server	kafka
KafkaRestMain	kafka-rest	confluent
SchemaRegistryMain	schema-registry	confluent

Shout-down

- Kill kafka Broker
- kill zookeeper
- reboot or:
  - rm -rf /tmp/kafka-logs
  - rm -rf /tmp/zookeper

Topic Utility

```
kafka-topic --create \
--bootstrap-server localhost:9092
--partitions 1
--replication-factor 1
--topic my-topic
```

Option	Parameter	Meaning
--bootstrap-server	<ip:port>	The broker list string in
create/delete		creates/deletes the topic
partitions	integer	number of partitions the topic has
replication-factors	integer	number of replicas each topic partition has
--topic	<topic-name>	

Console Producer

```
kafka-console-producer
--broker-list localhost:9092,localhost9093 \
    --property parse.key=true \
    --property key.separator=, \
    --topic testing
```

Option	Parameter	Meaning
--broker-list	<ip:port>	The broker list string in
--topic	<topic-name>	the form HOST1:PORT1,HOST2:PORT2.
--property	<String: prop>	user-defined properties in the form key=value

Console Consumer

```
kafka-console-consumer \
--bootstrap-server localhost:9092 \
--from-beginning \
--topic testing \
--property print.key=true
```

Option	Parameter	Meaning
--bootstrap-server	<ip:port>	The broker to enter the kafka cluster
--partition	integer	the partition to consume from
--from-beginning		sets the consumption from offset 0
--topic	<topic-name>	topic name
--property	<String: prop>	user-defined properties in the form key=value

Exercise 1

- create a topic “test”
- write some message to it using the console producer
- read those messages using the console consumer

Exercise 1b

- write some message on topic “test” **specifying a key** using the console producer
- read those messages using the console producer, **printing the key**

Exercise 2

- create a topic “test-2p” with two partitions
- write some message to it using the console producer
- read those messages using the console producer
- **What do you notice?**

Exercise 2b

- create a topic “test-2p” with two partitions
- write some message to it using the console producer
- read those messages using the console producer
- **consume the topic with two consumers, each assigned to a different partition**

Exercise 3

- create two topic “evens” and “odds”
- create a producer that writes even numbers to the homonymous topic
- create a consumer that reads from the topic “even” and prints to console
- BONUS:
  extend the consumer to sums 1 to each record and write to the topic “odds”.

Exercise 3

- create two topic “evens” and “odds”
- create a producer that writes even numbers to the homonymous topic
- create a consumer that reads from the topic “even” and prints to console

Exercise 3b

- extend the consumer to sums 1 to each record and write to the topic “odds”.
- re-run the consumer to obtain the augmented results.
- **What does happen?**

Exercise 3c

- By default kafka stores the consumer offset in a topic, considering the consumer group name.
- To start reading the topic from the beginning
  - change the consumer group name
  - programmatically seek for the beginning

Exercise 4

- Using console consumer, consume the topic “even” (or “odds” if you did the bonus point of exercise 3)

- **What does happen?**

Exercise 4b

- Using console consumer, consume the topic “even” (or “odds” if you did the bonus point of exercise 3)
- **You must be careful with serialisation and de-serialization**
- two properties for the console consumer:
- **key.deserializer**=org.apache.kafka.common.serialization.IntegerDeserializer
- **value.deserializer**=org.apache.kafka.common.serialization.IntegerDeserializer

Exercise 4

- Using console consumer, consume the topic “even” (or “odds” if you did the bonus point of exercise 3)
- **You must be careful with serialisation and de-serialization**
- two properties for the console consumer:
- **key.deserializer**=org.apache.kafka.common.serialization.IntegerDeserializer
- **value.deserializer**=org.apache.kafka.common.serialization.IntegerDeserializer