

## Lab 13-2 Custom Word Count



### Example

In this lab, you are going to run a custom Word Count Example. You must write the java code. You create two new Kafka topics, one called `word-count-input` and the other `word-count-output`. You will then add some data to the input topic, start a consumer on the output topic so you can watch it, and then run the Word Count Demo.

### Lab Solution

Complete solution for this lab is available in the following directory:

```
~/kafka-advanced/labs/Lab13-2
```

### Create Kafka Topics

- Create a topic named `word-count-input` with 1 partition and a replication factor of 1.
- Create a topic named `word-count-output` with 1 partition and a replication factor of 1.

**ACTION** - REVIEW `create-topics.sh`

`~/kafka-advanced/labs/Lab13-2/bin/create-topics.sh`

```
#!/usr/bin/env bash
cd ~/kafka-advanced

## Create input topic
kafka/bin/kafka-topics.sh --create \
  --replication-factor 1 \
  --partitions 1 \
  --topic word-count-input \
  --zookeeper localhost:2181

## Create output topic
kafka/bin/kafka-topics.sh --create \
  --replication-factor 1 \
  --partitions 1 \
  --topic word-count-output \
  --zookeeper localhost:2181

## List created topics
kafka/bin/kafka-topics.sh --list \
  --zookeeper localhost:2181
```

**ACTION** - START ZooKeeper and Kafka Broker if needed. Only 1 broker is necessary.

**ACTION** - RUN `create-topics.sh` as follows:

```
$ cd ~/kafka-advanced/labs/Lab13-2/bin
$ ./create-topics.sh
Created topic "word-count-input".
Created topic "word-count-output".
```

```
word-count-input
word-count-output
```

## Add data to the topic

Now add data to the input topic by starting a console producer and entering data.

**ACTION** - REVIEW `start-producer-console-input.sh`

`~/kafka-advanced/labs/Lab13-2/bin/start-producer-console-input.sh`

```
#!/usr/bin/env bash
cd ~/kafka-advanced

## Producer
kafka/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic streams-plaintext-input
```

**ACTION** - RUN `start-producer-console-input.sh`

```
$ cd ~/kafka-advanced/labs/Lab13-2
$ ./start-producer-console-input.sh
>
```

## ACTION - Enter Data

Enter the following 3 lines of data. Ctrl-C to stop the console.

```
> stock streams MSFT
> stock data AAPL
> stock streams RHT
```

## ACTION - Validate Data

Run a console consumer against the input topic to verify the data is there.

`~/kafka-advanced/labs/Lab13-2/start-consumer-console-input.sh`

```
#!/usr/bin/env bash
cd ~/kafka-advanced

## Input Consumer
kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic streams-plaintext-input --from-beginning
```

**ACTION** - RUN `start-consumer-console-input.sh`

```
$ cd ~/kafka-advanced/labs/Lab13-2/bin
$ ./start-consumer-console-input.sh
> stock streams MSFT
> stock data AAPL
> stock streams RHT
```

```
<Ctrl-C>
Processed a total of 3 messages
```

Above, stops the consumer.

## Run a console consumer against the output.

Run another consumer against the output topic so we can see the work that the Word Count Demo does. Notice a few things about this consumer. These match the way the topic is populated by the demo.

- It has a message formatter specified.
- It has key and value deserializers.

**ACTION** - EDIT `~/kafka-advanced/labs/Lab13-2/bin/start-consumer-console-output.sh`, follow instructions in file.

**~/kafka-advanced/labs/Lab13-2/start-consumer-console-output.sh**

```
#!/usr/bin/env bash
cd ~/kafka-advanced

## Output Consumer
kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 \
  --topic word-count-output \
  --from-beginning \
  --formatter kafka.tools.DefaultMessageFormatter \
  --property print.key=true \
  --property print.value=true \
  --property
key.deserializer=org.apache.kafka.common.serialization.StringDeserializer \
  --property
value.deserializer=org.apache.kafka.common.serialization.LongDeserializer
```

**ACTION** - RUN `start-consumer-console-output.sh`

```
$ cd ~/kafka-advanced/labs/Lab13-2/bin
$ ./start-consumer-console-output.sh
```

There will be no output until the demo code runs. Keep this window open so you can see it as the demo runs.

## Create and Run the Custom Word Count

**ACTION** - EDIT `~/kafka-advanced/labs/Lab13-2/WordCount.java`, follow instructions in file.

**ACTION** - RUN `WordCount` Run the java main.

**Consumer Console Output after running run-word-count-demo.sh**

```
stock 1
streams 1
msft 1
stock 2
data 1
```

```
aapl    1
stock   3
streams 2
rht     1
```

### **Conclusion Custom Word Count example**

The word count is the simplest example of stream processing of an input topic and writing results to a different then you created Kafka Producer in Java that uses the Kafka replicated topic to send records. You sent records with the Kafka Producer using async and sync send methods.

### **Review Custom Word Count example**

#### **How could you change the code to not be case-insensitive?**

Remove the `.toLowerCase()` from step 2

#### **How would you simplify the code by using the builder syntax?**

Chain calls together