**Lab 01: Kafka Basics**

# **Introduction**

In this lab, you will explore Kafka basics. You will utilize command-line tools provided by Confluent to start up Kafka and related components. Then you will create a Kafka topic and use the producer and consumer to utilize the messages written to the topic.

# **Let’s get Started**

The Confluent platform provides a single script that starts the following components:

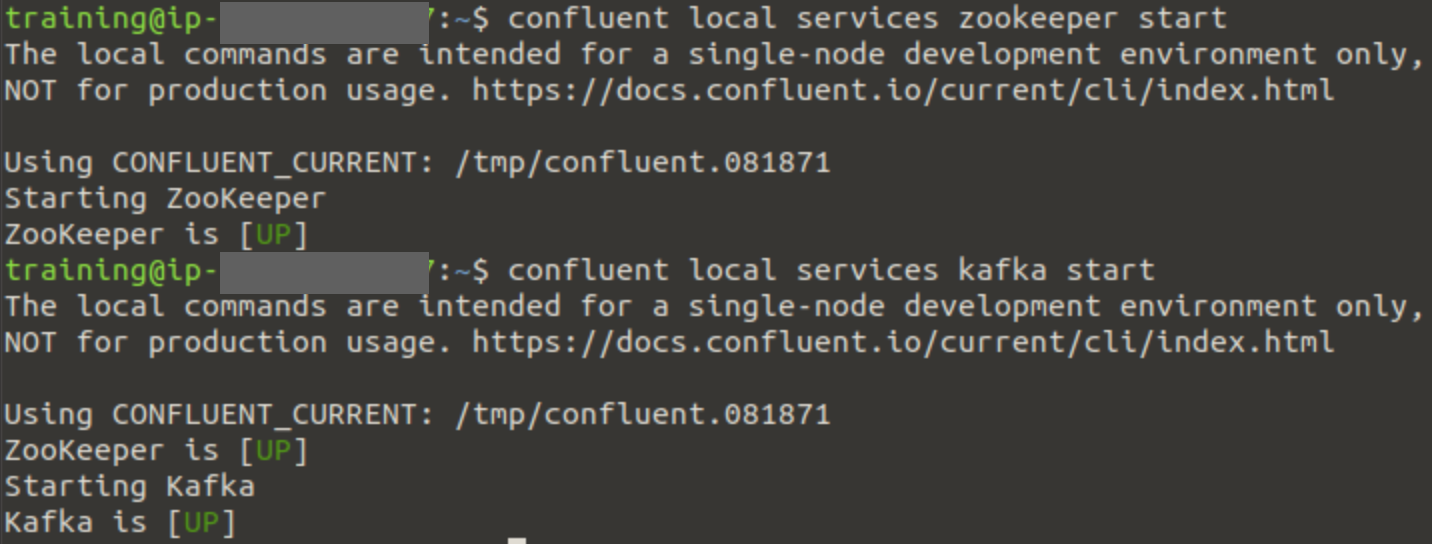
* Kafka,
* Zookeeper
* Schema Registry
* Kafka-rest
* Connect
* KSQLDB Server
* Control Center

The script exists in the confluent bin directory. We will use the confluent script and start them all at once:

1. Open Terminal window
2. Execute the following command to start confluent Kafka and related components:

| confluent local services zookeeper start confluent local services kafka start |
| --- |

You will see the following content in the terminal window:



1. Keep the Terminal window open.

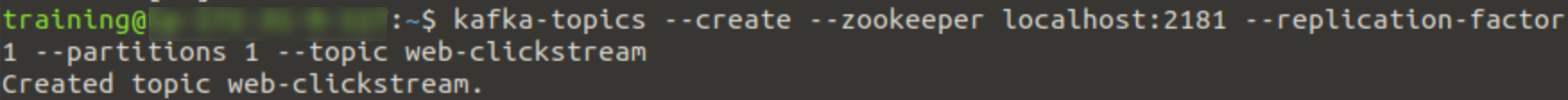
## Create a Kafka Topic

Kafka maintains feeds of messages in categories called topics. In this part, you will create a topic named "web-clickstream" with a single partition and only one replica.

1. Open a new **Terminal** window by clicking **Application > Terminal.**
2. Check the current directory using the **'pwd'** command.
3. Execute the following command to create a Kafka topic:

| kafka-topics --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic web-clickstream |
| --- |

The console output should show:



1. Show a list of Kafka topics:

| kafka-topics --list --zookeeper localhost:2181 |
| --- |

In the last, we can see our topic “web-clickstream”

## Send Some Messages

Kafka comes with a command-line client that will take input from a file or from standard input and send it out as messages to the Kafka cluster.

1. In the Producer terminal window, execute the following command to send some messages to the Kafka cluster:

| kafka-console-producer --broker-list localhost:9092 --topic web-clickstream |
| --- |

Notice it shows you a prompt ‘>’ where you can start entering messages which will be sent to the Kafka cluster.

1. Enter the following messages at the prompt. Hit the enter key after each message:

**hello world**

**web-clickstream 1**

**web-clickstream 2**

Don't exit the > prompt. You can exit the prompt by pressing Ctrl+Z, but don't do it until instructed.

1. Keep the Terminal window open.

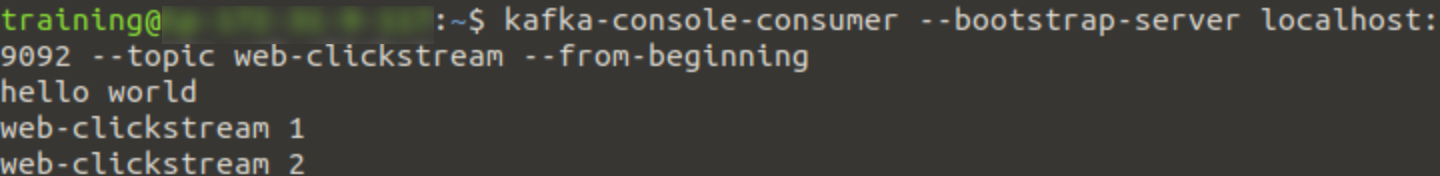
## Read messages from the Kafka cluster

Kafka also has a command-line consumer that will dump out messages to standard output.

1. Open a new Terminal window
2. Enter the following command to read the messages:

| kafka-console-consumer --bootstrap-server localhost:9092 --topic web-clickstream --from-beginning |
| --- |

1. Notice that after few seconds it shows the following messages:



1. Switch back to the **Producer Terminal** window and enter a few more messages.
2. Switch back to the **Consumer Terminal** window and notice that the newly entered messages automatically show up!
3. Switch to the **Producer Terminal** window and press **Ctrl + Z** to stop entering more messages. Keep the Terminal window open.
4. Switch to the **Consumer Terminal** window and press **Ctrl + Z** to stop reading messages. Keep the Terminal window open.

# Messages with key-value pairs

## Produce records with key-value pairs

Kafka lets you send keys and values, but so far you’ve only sent records with values. To be fair, you’ve sent key-value pairs, but the keys are null. Sometimes you’ll need to send a valid key in addition to the value from the command line.

For sending full key-value pairs from the command line there is the requirement to add two properties to console producer, parse.key and key.separator

Let’s try to send some full key-value records now. If your previous console producer is still running close it with a CTRL+C and run the following command to start a new console producer:

| kafka-console-producer \  --topic web-clickstream \  --bootstrap-server localhost:9092 \  --property parse.key=true \  --property key.separator="|" |
| --- |

Enter the following messages at the prompt. Hit the enter key after each message:

**1 | hello world**

**2 | web-clickstream 1**

**one|bar**

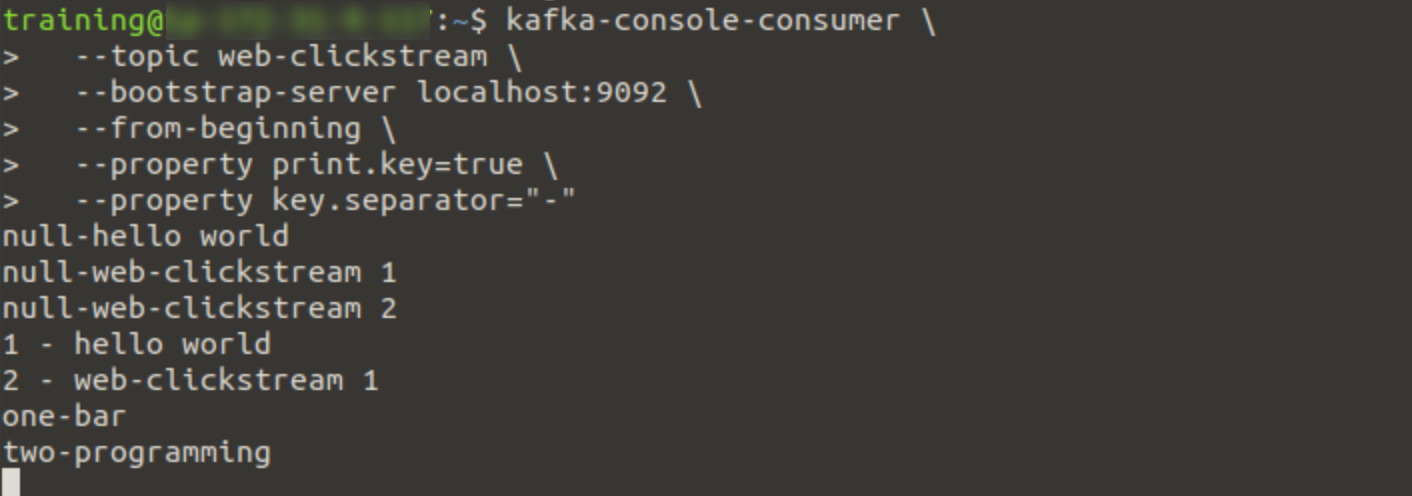
**two|programming**

# 

## Start a consumer to show full key-value pairs

Now that we’ve produced full key-value pairs from the command line, you’ll want to consume full key-value pairs from the command line as well. Close the previous console consumer with a CTRL+C and run the following command: print.key Prints out a portion of the key-value pair data in your terminal window

| kafka-console-consumer \  --topic web-clickstream \  --bootstrap-server localhost:9092 \  --from-beginning \  --property print.key=true \  --property key.separator="-" |
| --- |

****

Since we kept the **--from-beginning** property, you’ll see all the records sent to the topic. You’ll notice the results before you send keys that are formatted as **null-<value>**.

# 

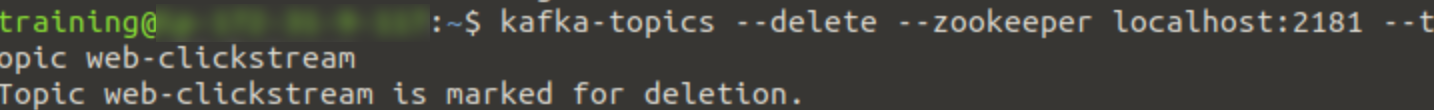
# Deleting Kafka Topic

In this part, you will delete the Kafka topic which you previously created in this lab.

1. In the Producer Terminal window, run the following command.

| kafka-topics --delete --zookeeper localhost:2181 --topic web-clickstream |
| --- |

This should show in the console output:



1. Use the following command to verify that the 'web-clickstream' topic doesn't exist anymore:

| kafka-topics --list --zookeeper localhost:2181 |
| --- |

If the topic is not deleted go back and make sure that you closed the producer and consumer prompts. Then retry the delete and check again.

1. Switch to the first Terminal and stop the confluent Kafka:

| confluent local services stop |
| --- |

**Voila!!** We have successfully published and consumed messages using Kafka Console.