WeCanDoNow HandsOn Lab - Kafka Streams API

by Saravanan Sundaramoorthy

The Streams API allows transforming streams of data from input topics to output topics.

```
sudo apt install maven -y

mvn archetype:generate -DgroupId=com.wecandonow -DartifactId=WeCanDoNow-
Kafka-Streams -DarchetypeArtifactId=maven-archetype-quickstart
cd WeCanDoNow-Kafka-Streams/
```

Add Kafka dependencies in `pom.xml`

```
cd WeCanDoNow-Kafka-Streams

vim pom.xml
```

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.wecandonow</groupId>
  <artifactId>Kafka-Producer</artifactId>
  <packaging>jar</packaging>
  <version>1.0-SNAPSHOT</version>
  <name>Kafka-Producer</name>
  <url>http://maven.apache.org</url>

  <build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.7.0</version>
            <configuration>
                <source>1.8</source>
                <target>1.8</target>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-assembly-plugin</artifactId>
            <version>2.4.1</version>
```

```
            <configuration>
                <archive>
                    <manifest>
                        <mainClass>com.wecandonow.App</mainClass>
                    </manifest>
                </archive>
                <descriptorRefs>
                    <descriptorRef>jar-with-dependencies</descriptorRef>
                </descriptorRefs>
                <skipAssembly>false</skipAssembly>
            </configuration>
            <executions>
                <execution>
                    <id>package</id>
                    <phase>package</phase>
                    <goals>
                        <goal>single</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
  </build>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>

    <dependency>
        <groupId>org.apache.kafka</groupId>
        <artifactId>kafka_2.12</artifactId>
        <version>1.0.1</version>
    </dependency>

    <dependency>
        <groupId>org.apache.kafka</groupId>
        <artifactId>kafka-clients</artifactId>
        <version>1.0.1</version>
    </dependency>

    <dependency>
        <groupId>org.apache.kafka</groupId>
        <artifactId>kafka-streams</artifactId>
        <version>1.0.1</version>
    </dependency>
  </dependencies>
</project>
```

Streams.java

```
vim src/main/java/com/wecandonow/Streams.java
```

Reads from input topic and sends message length to ouput topic

```
package com.wecandonow;

import org.apache.kafka.common.serialization.Serdes;
import org.apache.kafka.streams.KafkaStreams;
import org.apache.kafka.streams.StreamsBuilder;
import org.apache.kafka.streams.StreamsConfig;

import java.util.HashMap;
import java.util.Map;

public class Streams
{
    public static void main(String[] args) {
        if (args.length != 2) {
            System.out.println("Usage: <input_topic> <output_topic>");
            return;
        }

        Map<String, Object> props = new HashMap<>();
            props.put(StreamsConfig.APPLICATION_ID_CONFIG, "my-stream-
processing-application");
            props.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG,
"localhost:9092");
            props.put(StreamsConfig.DEFAULT_KEY_SERDE_CLASS_CONFIG,
Serdes.String().getClass());
            props.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG,
Serdes.String().getClass());
            StreamsConfig config = new StreamsConfig(props);

            StreamsBuilder builder = new StreamsBuilder();
            builder.<String, String>stream(args[0]).mapValues(value ->
value.length() + "").to(args[1]);

            KafkaStreams streams = new KafkaStreams(builder.build(),
config);
            streams.start();
    }
}
```

Create input and output topics

```
kafka-topics.sh --delete --zookeeper localhost:2181 --topic input-topic
kafka-topics.sh --delete --zookeeper localhost:2181 --topic output-topic

kafka-topics.sh --create --zookeeper localhost:2181 --topic input-topic --
partitions 1 --replication-factor 1
kafka-topics.sh --create --zookeeper localhost:2181 --topic output-topic -
-partitions 1 --replication-factor 1
```

Compile and Run the App

```
mvn compile
mvn exec:java -Dexec.mainClass="com.wecandonow.Streams" -
Dexec.args="input-topic output-topic"
```

Output

```
kafka-console-producer.sh --broker-list localhost:9092 --topic input-topic
>hello
>world
>wecandonow
```

```
kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic
output-topic --from-beginning
5
5
7
```