# **INTEGRATING KAFKA AND IGNITE**

The final part of the project brings together the different components worked on in the previous sections. This involves loading the Messages generated in Part 1 into a CSV file, publishing each of these from the file to a UDP port. Kafka then reads from this port and stores it in its broker. This data is finally streamed to Ignite cache via KafkaConnect and queries are performed on it.

\*\* Parts of the documentation that are written in this color have not been implemented. They are guidelines to improve on the existing work.

## **Working with CSV**

**import com.opencsv.\*;**

This package is used to perform all CSV-related operations in Java.

**FileWriter** and **FileReader** can be used to write to and read from the file respectively. For writing, refer **MakeCSV.java** and for reading, **WriteToUDP.java**.

## **Publishing to UDP**

**WriteToUDP.java** contains the code that publishes to a specific UDP port. In this case, the port used is 35352.

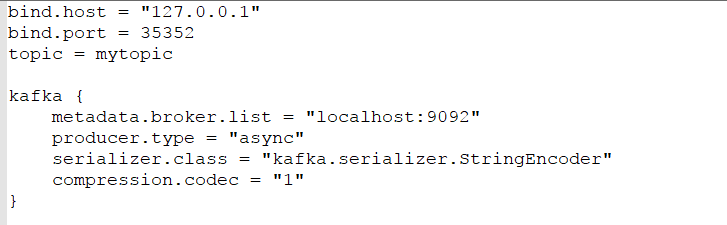
To successfully publish, the data cannot be sent directly. It has to be serialized by converting it into an array of bytes. With strings, the method is very straightforward as demonstrated in the code.

To serialize an object of a class, these methods will have to be implemented in the respective class -



## **Streaming from UDP to Kafka**

The JAR file, **udp-kafka-bridge-assembly-0.1.jar** accomplishes this task. The configurations for this streaming can be stored in **bridge.conf** and configured as follows -



Where, bind.host – localhost (or 127.0.0.1).

bind.port – port to read from (35352 in this case).

bind.topic – topic into which data is streamed.

kafka.serializer.class - to be modified if data other than that of String type has to be streamed.

To successfully stream data, run this JAR using the following command -

**java -Dconfig.file=bridge.conf -jar udp-kafka-bridge-assembly-0.1.jar**

Open a new terminal and run the program that is publishing to the port. This should stream data directly to the Kafka topic mentioned in **bridge.conf**.

This JAR was obtained from the GitHub repository, <https://github.com/agaoglu/udp-kafka-bridge>.

## **From Kafka to Ignite**

This is covered in Part 2 of the project.

Some points to be considered are -

* The Kafka topics streamed to from the UDP port should be same as the topics Ignite consumes data from.
* Cache where data is stored should be configured beforehand, and indexing must be enabled.
* The key, value types of Kafka should match that of Ignite.

## **The Data in Ignite**

The data that is loaded into Ignite can be viewed in one of 2 ways – by running queries on the cache, or using the REST API.

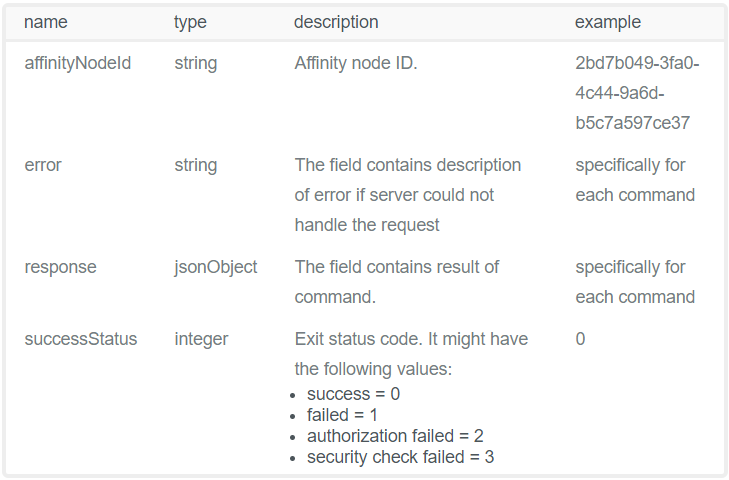
**Enabling REST**

* The required files to start REST are present in the directory, ignite-rest-http, which is initially in apache-ignite-fabric-2.0.0-bin/libs/optional/.
* Move the directory into apache-ignite-fabric-2.0.0-bin/libs/.
* Start the Ignite instance, and REST should be running.

To test it, run the following command -

**curl http://localhost:8080/ignite?cmd=version**

This should return the version of Ignite. The return value is of the following format -



For more commands, refer <https://apacheignite.readme.io/v1.3/docs/rest-api>.