

# Gigaspaces Kafka Sink Connector

## Prerequisite

- Installation of Kafka and Kafka Connect
- [Installation of Insightedge v15.2](#)
- Git, Maven and JDK 8

## Install

- git clone this repo
- mvn clean package
- Move the generated jar (from the target folder) to the kafka connect connectors lib folder
- Define the connector configuration as outlined below
- Schema and type definitions for the data can be expressed via the json file as shown below.

**Note:** The maven Kafka artifacts in the pom.xml file must match the Kafka version.

**Note:** If you have developed a GigaSpaces data model, you do not have to provide a json file. Instead, you can provide the generated jar file containing the relevant POJOs.

## Configuration

### Gigaspaces connector properties file example (connect-gigaspaces-sink.properties):

```
bootstrap.servers=localhost:9092
name=gigaspaces-kafka
connector.class=com.gigaspaces.kafka.connector.GigaspacesSinkConnector
tasks.max=1
topics=Pet,Person
gs.connector.name=gs
# True -- start gs inside the same JVM as connector; False - separate JVM (default)
gs.space.embedded=false
# Name of the target gs Space
gs.space.name=demo
# Location of GS Manager:
gs.space.locator=127.0.0.1:4174
#Choose one of the following -- Jar file or Json file:
gs.model.json.path=<path to gigaspaces kafka connector repo>/example/resources/model.json
#
plugin.path=<path to gigaspaces kafka connector repo>

value.converter=org.apache.kafka.connect.json.JsonConverter
value.converter.schemas.enable=false

key.converter=org.apache.kafka.connect.storage.StringConverter
# Currently the connector does not support Kafka schema.
key.converter.schemas.enable=false
#key.converter.schemas.enable=true
#value.converter.schemas.enable=true

offset.storage.file.filename=/tmp/connect.offsets
# Flush much faster than normal, which is useful for testing/debugging
offset.flush.interval.ms=10000
```

## Gigaspaces connector model schema json file example

**Note:** These Json fields map to the Space Type Descriptor in GS. For more information, see [Space type Descriptor](#) in the GigaSpaces documentation center.

```
[{
  "type": "com.gs.Person",
  "FixedProperties": {
    "firstname": "java.lang.String",
    "lastname": "java.lang.String",
    "age": "java.lang.Integer",
    "num": "java.lang.Integer"
  },
  "Indexes": {
    "compoundIdx": {"type":"EQUAL", "properties": ["firstname", "lastname"], "unique": false},
    "ageIdx": {"type":"ORDERED", "properties": ["age"], "unique": false}
  },
  "Id": "num",
  "RoutingProperty": "firstname"
},
{
  "type": "com.gs.Pet",
  "FixedProperties": {
    "kind": "java.lang.String",
    "name": "java.lang.String",
    "age": "java.lang.Integer"
  },
  "Indexes": {
    "compoundIdx": {"type":"EQUAL", "properties": ["kind", "name"], "unique": false},
    "ageIdx": {"type":"ORDERED", "properties": ["age"], "unique": false}
  },
  "RoutingProperty": "name"
}]
```

## Running the example:

**Note:** The steps must be run in the order indicated below.

In this example, we will consume data from a text file using the FileStreamSource source connector. This connector will publish the lines it reads to the type topics in Kafka. The Gigaspaces sink connector will read the data from the topics and store them in the in-memory grid (the "Space"). All files are under the example/resources folder.

1.Start Gigaspaces and have a Space running. In this example, we are running the demo project: `gs.sh demo`

2.Start Zookeeper.

**Note:** Do not use port 2181.

3.Start Kafka using the same port used for Zookeeper.

4.Start the connect with the source and sink connectors and see how the data is consumed and published to the space:

```
connect-standalone connect-standalone.properties people-source.properties pet-
```


source.properties connect-gigaspace-sink.properties

**Note:** The three connectors properties are found in <path to gigaspaces kafka connector repo>/example/resources.

**Note:** Ensure that the file parameter in the people-source.properties file and the pet-source.properties file points to the location of the corresponding txt files.

5. Connect to the gigaspaces UI and view the types that were defined and the data that was inserted into the spaces by the connector.

1.

From the Ops Manager screen, choose **Analyze Ops Manager:** 

2.

In the Spaces Overview, select the **demo** Space:



3. You can now see the two object types, **Pet** and **Person**, and the number of entries for each object:

