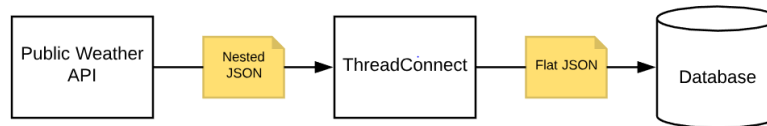# MAPPER (JSON TRANSFORMATION)

To transform nested JSON (response of Weather API) into a flat JSON using ThreadConnectMapper. Specific fields of flat JSON is inputted as a record into Postgres DB.

Created by K Sanghavi

KEY ATTRIBUTES

- `InvokeHttp`: invokes the weather API endpoint
- `ThreadConnectMapper`: transforms incoming JSON data into required JSON format using jolt-specs.
- `EvaluateJsonPath`: extracts specified values from the incoming JSON based on the path.
- `ReplaceText`: creates SQL statement with extracted flowfile attributes.
- `PutSQL`: executes the SQL command on the specified (postgres) database. PutSQL inputs the record passed as data into "test" table.

CONTROLLER SERVICES USED

- `StandardSSLContextService`: helps to communicate with secure API endpoint(HTTPS)
- `DBCPConnectionPool`: connects to the specified postgres database

MAPPING USING THREADCONNECTMAPPER

1. To view the Advance UI, click on ADVANCED button at the bottom left corner of the processor's configuration dialog.

2. ThreadConnectMapper Advance UI view

3. Based on the type of transformation (JSON-JSON), select the Source & Target type and upload Source and Target input file using Choose File button. (extensions supported for JSON - .json). If the loaded file content for source and target has no syntax errors (if present, corresponding line will be highlighted), then click on Map tab.

4. The Map tab represents the source and target data as collapsible tree structure. To create a mapping drag a line from the required source element and drop at the target element. To scroll, scrollbar is provided for both Source and Target. Mapping can be deleted by double clicking on the specific mapping. If all the mappings has to be deleted then by clicking on Reset button, mappings created get deleted.
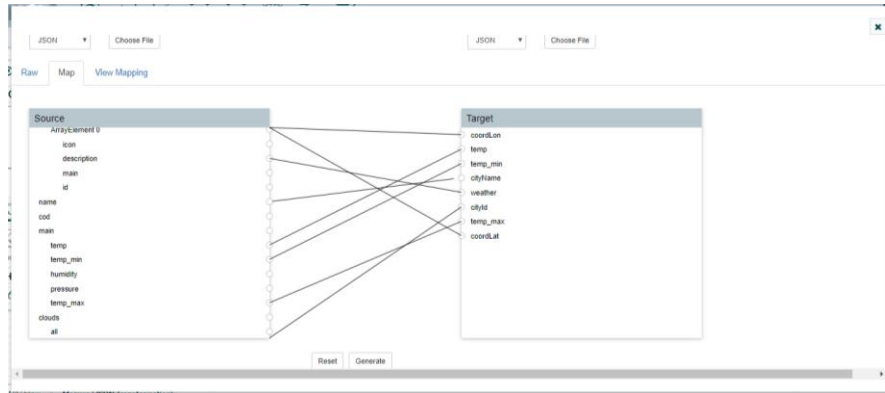
5. On mapping the required elements, click on the Generate button. This generates the required jolt-spec based on the mapping. Which can be viewed in the ViewMapping tab.
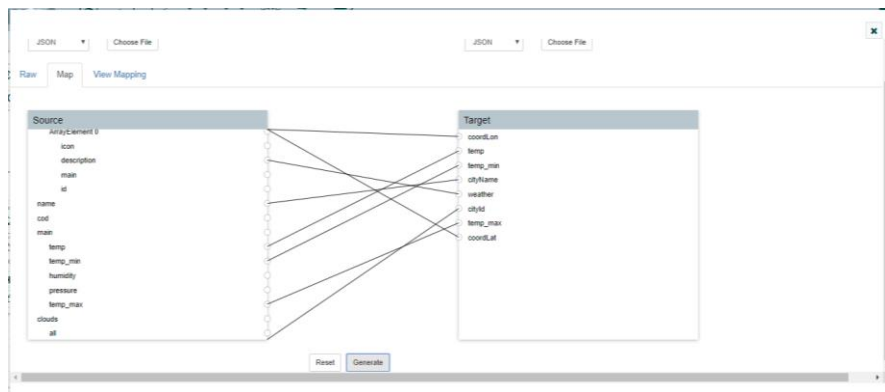
6. To update the ThreadConnectMapper with the generated spec click on Update Processor button. On successful operation, user is prompted with a success message.
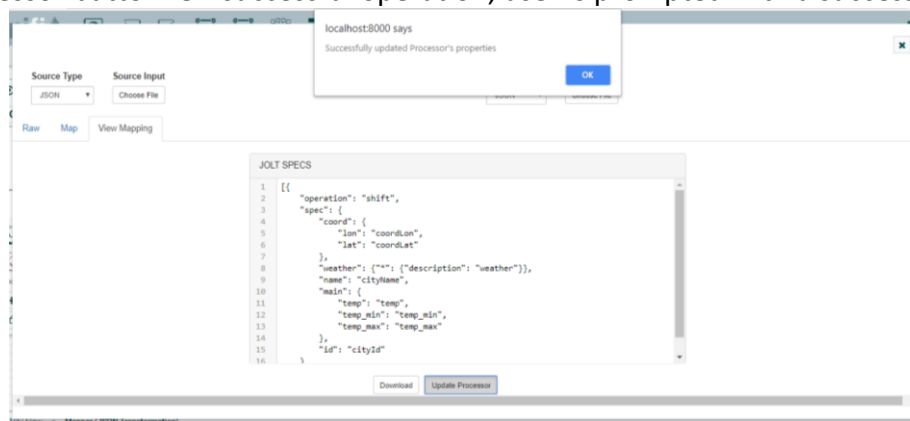
7. Properties Tab

NOTES

- The template requires Postgres DB with following requirements:

  DB Name: postgres

  User: postgres

  Password: postgres

  Table Name: test

  Schema:

  {

  "column name" : "type"

  id : character varying,

  cityname : character varying,

  citytemp : character varying,

  }

  Database Driver Location : Postgressql JDBC Driver jar location (download from https://jdbc.postgresql.org/download.html)

- Carefully check that database configuration details are updated across the **entire** flow when configuring a new database connection.