iWay Software

How-to's

# Using the JSON Iterator

This topic describes how to process a JSON document, which contains multiple records. A JSON document will be split into sub-documents using the JSON Iterator, and then each sub-document will be sent for further processing. This is useful whenever a request is received containing multiple records that are batched into a single JSON document. This how-to will also demonstrate how you can test the process by providing JSON as an input document. This will simulate receiving JSON as an inbound data document from any supported protocol. Note that this how-to does not implement error handling, which should be implemented in production applications.

**Included With This How-To:**

The following sample components are included with this how-to in the *json_iterator_sample_files.zip* file:

- **iterator_project_json.zip** – A completed application project for this how-to that you can import into iWay Integration Tools (iIT) using the *Import Existing Project into Workspace* option.  After importing, you can browse the components and proceed to deploying the application.

- **po_item10.json** – A sample JSON input document that is referenced by this how-to.

This how-to includes the following topics:

## Prerequisites

Before continuing, ensure that the following prerequisites are confirmed.

- **Folder.** The following output folder is predefined for the File listener:

  ```
  C:\temp\out
  ```

  Ensure that this folder created before proceeding with this how-to. However, you may also define other paths for the output folder when configuring the File listener.

- **JSON File (po_item10.json).** A sample JSON file containing multiple records, which is provided with this how-to.

- **iWay Service Manager.** A running instance of iWay Service Manager (iSM).

## Summary of Steps and Key Features

This how-to consists of the following steps:

1. Creating an application project.

2. Creating a process flow, which will take the JSON file, iterate it through each record in the file, and then output it.

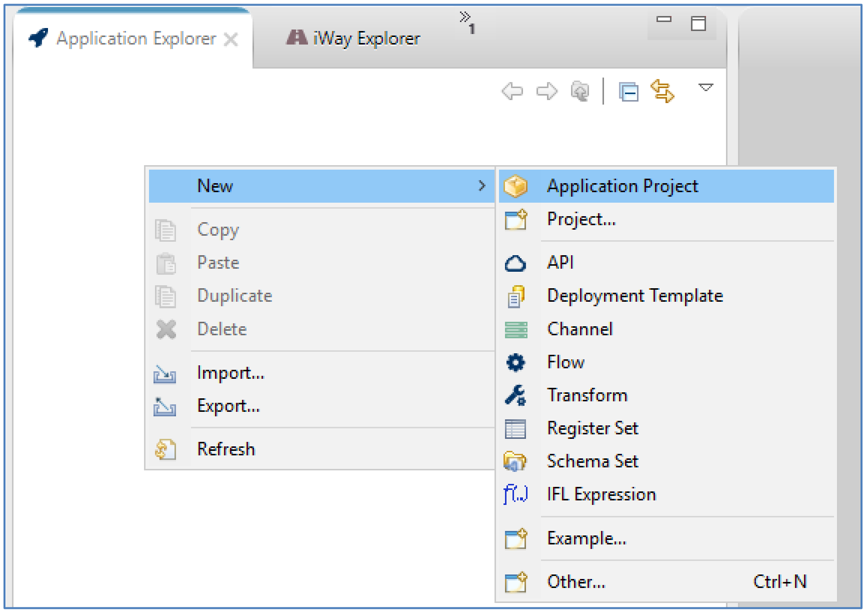
The following key features are demonstrated by this how-to:

- JSON Iterator
- File Write Object
- Process Flow Test-Run Operation

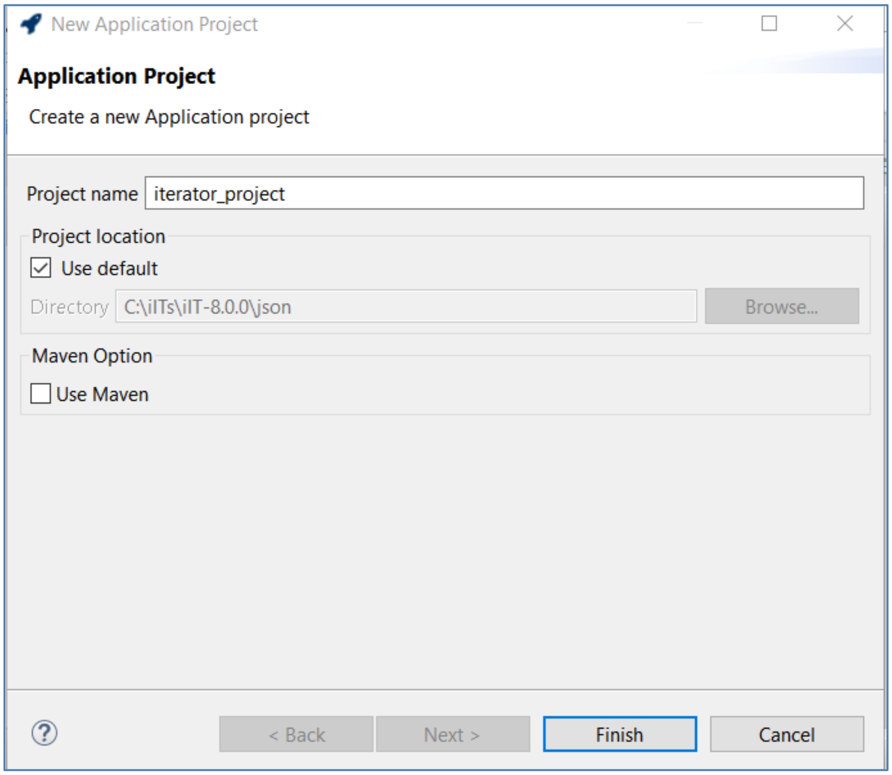
## Creating an Application Project

1. Open iWay Integration Tools (iIT) and select the default workspace.

2. Right-click anywhere within the Application Explorer tab, select *New* from the context menu, and then click *Application Project*, as shown in the following image.



The New Application Project dialog opens, as shown in the following image.



Application Project

Create a new Application project

Project name: iterator_project

Project location

☑ Use default

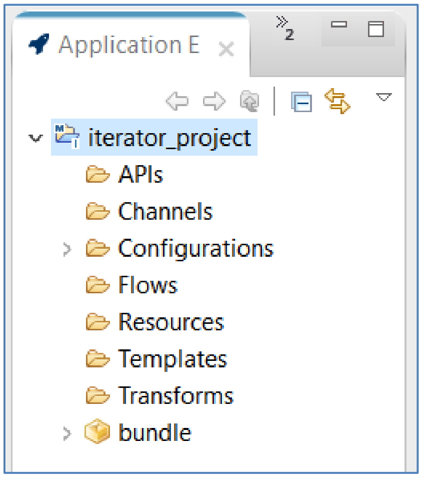
Directory: C:\iITs\iIT-8.0.0\json

Maven Option

□ Use Maven

3. Provide a project name (for example, *iterator_project*) and then click *Finish*.

   A new application is created, containing the required project folder structure, as shown in the following image.

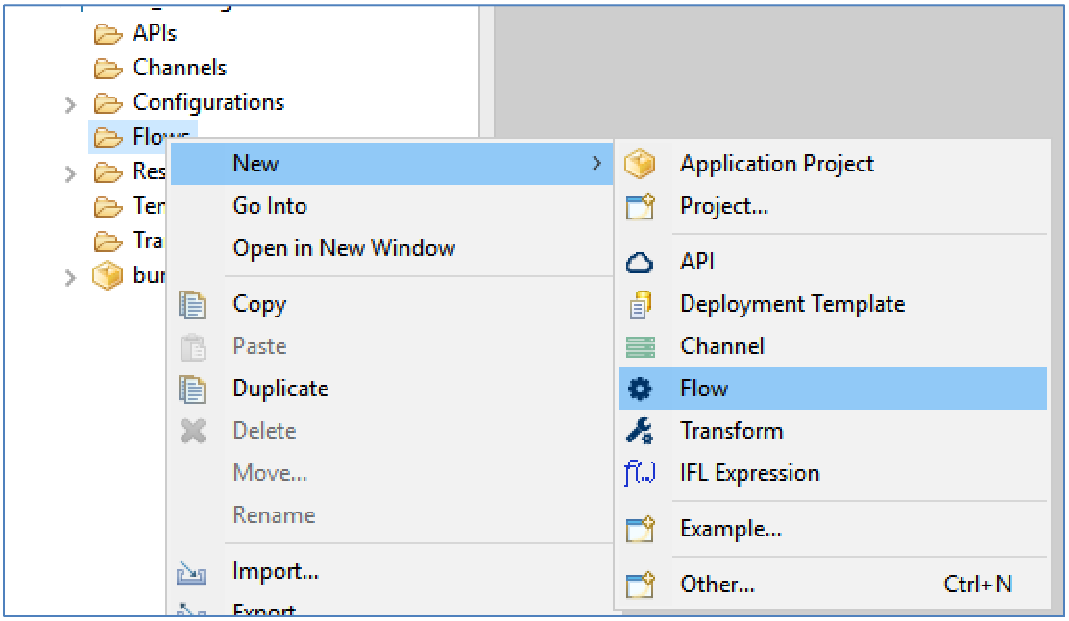


The *bundle* folder is the application package name, which you can rename if required. During deployment, you will be prompted to select the name of the deployed application.

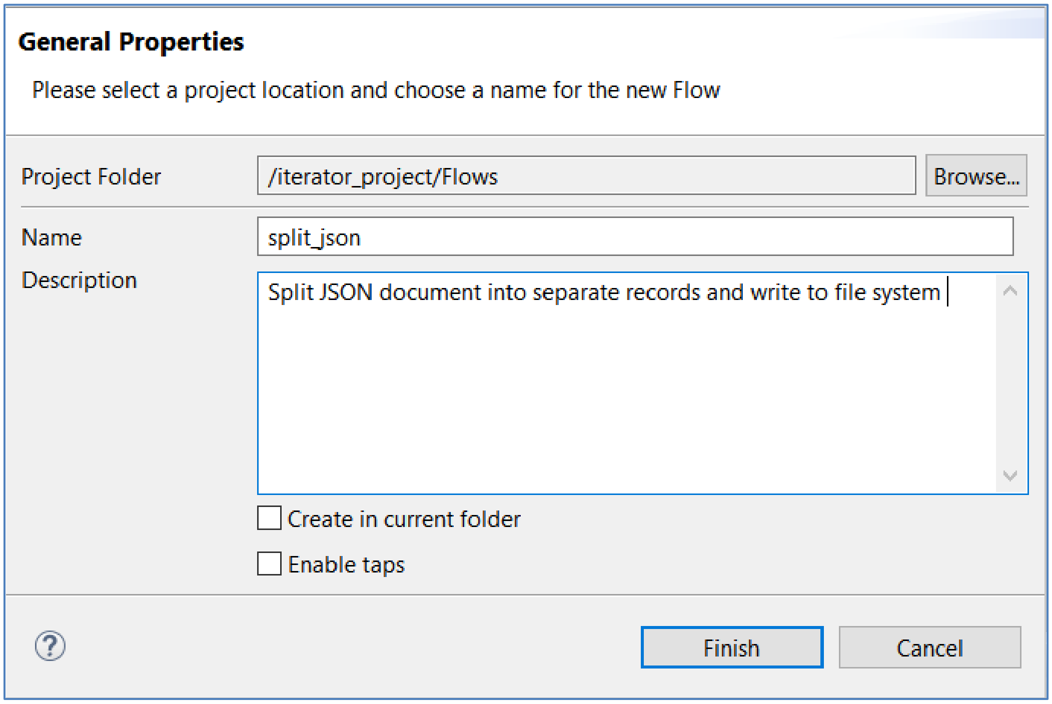
You are now ready to configure your process flow.

## Configuring Your Process Flow

1. Right-click the *Flows* folder, select *New* and then click *Flow*, as shown in the following image.



The General Properties dialog opens, as shown in the following image.



**General Properties**

Please select a project location and choose a name for the new Flow

Project Folder: /iterator_project/Flows

Name: split_json

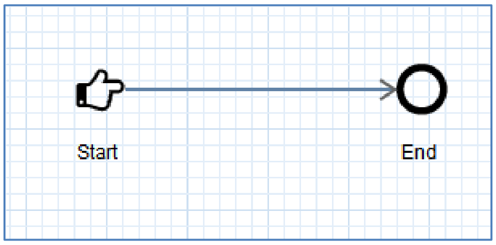
Description: Split JSON document into separate records and write to file system

□ Create in current folder

□ Enable taps

2. Enter a name for your process flow (for example, *split_json*), a description (optional), and then click *Finish*.

   The process flow (*split_json)* opens as a new tab in your workspace area where you can start building the application logic for processing data. By default, the process flow consists of a Start and an End object. You can also drag the End object to the right to give yourself more room to work with.

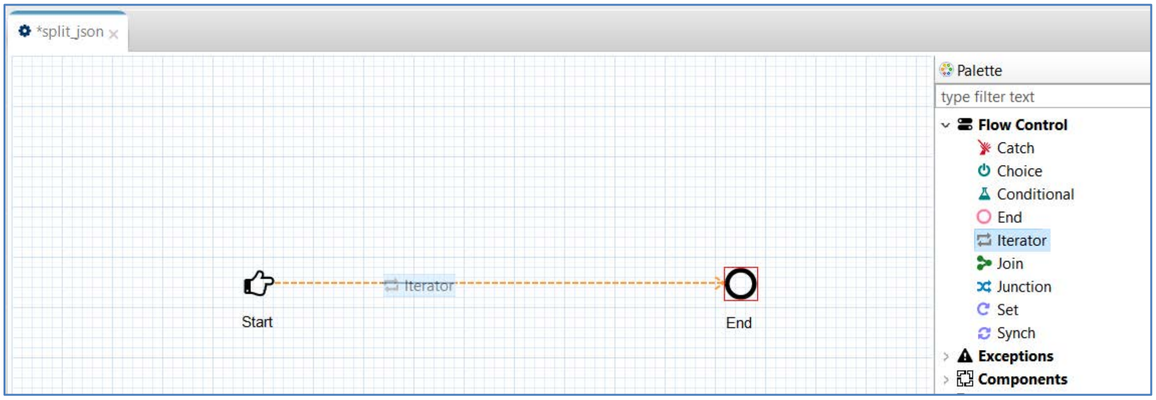


   In this how-to, you will configure this process flow to receive a JSON input document containing multiple records. By adding an Iterator to the process flow, you will split the JSON document into many JSON sub-documents for additional processing.

## Adding an Iterator Object

1. From the Palette in the right pane, expand *Flow Control* and drag the *Iterator* object onto the line between the Start and End objects in your workspace area.

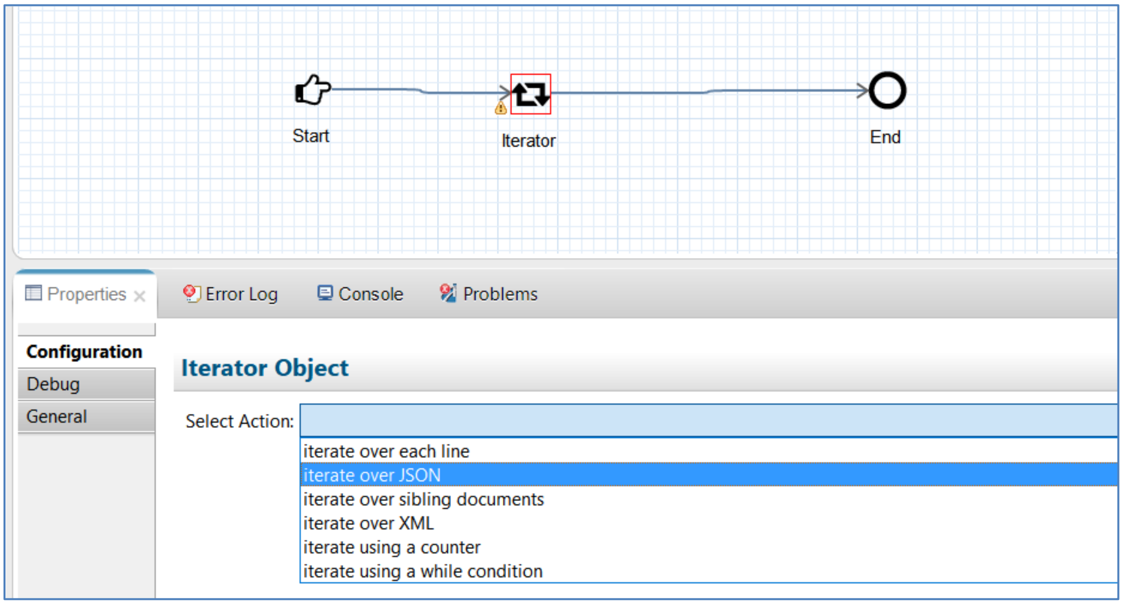
   When the solid connector line between the objects changes to a dashed orange line, release the Iterator object, as shown in the following image.



You can also drag the Iterator object onto any location in the process flow designer to be configured and then recreate the connector line relationships as required.

Once the Iterator object is added to the process flow, the corresponding configuration properties for this object are available in the Properties tab.

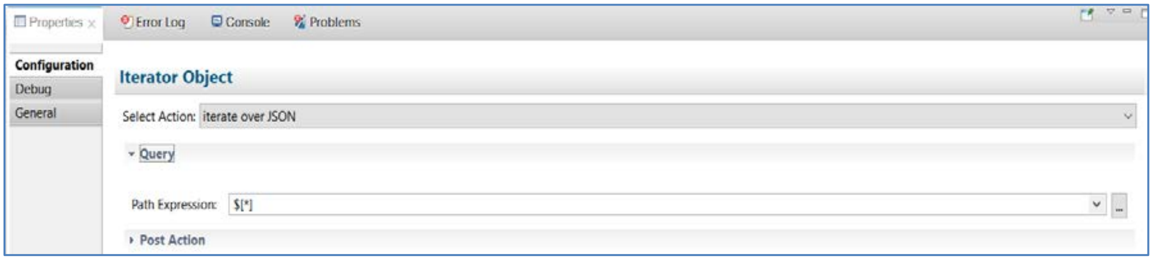
2. From the Select Action drop-down list, select *Iterate over JSON*, as shown in the following image.



A list of configuration properties for this action is provided. The only required property is *Query*, which specifies what to use for parsing out the records.

3. Enter the following value in the Path Expression field:

```
$[*]
```



Iterator Object

Select Action: iterate over JSON

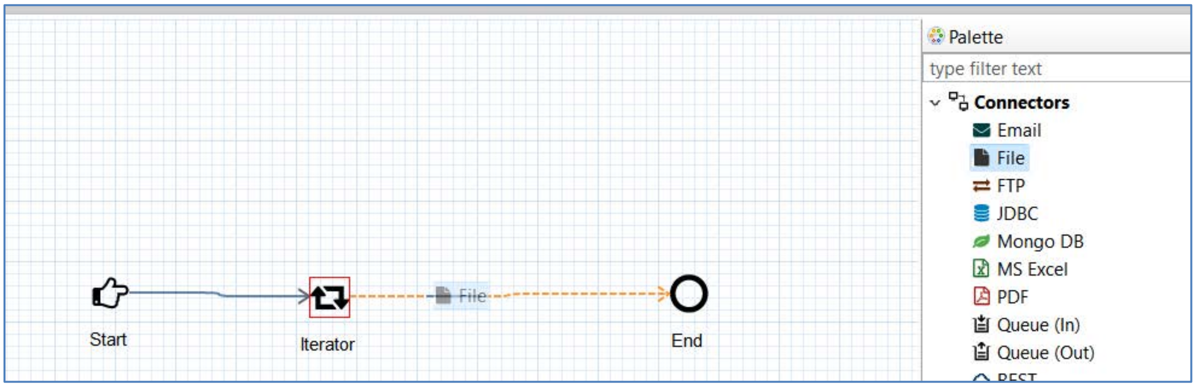
Path Expression: $[*]

You are now ready to specify the next step for each record to take. This can be anything based on the application requirements, such as executing a REST call, performing further transformations, or validations, and so on. For this how-to, you will write out each record to the file system.

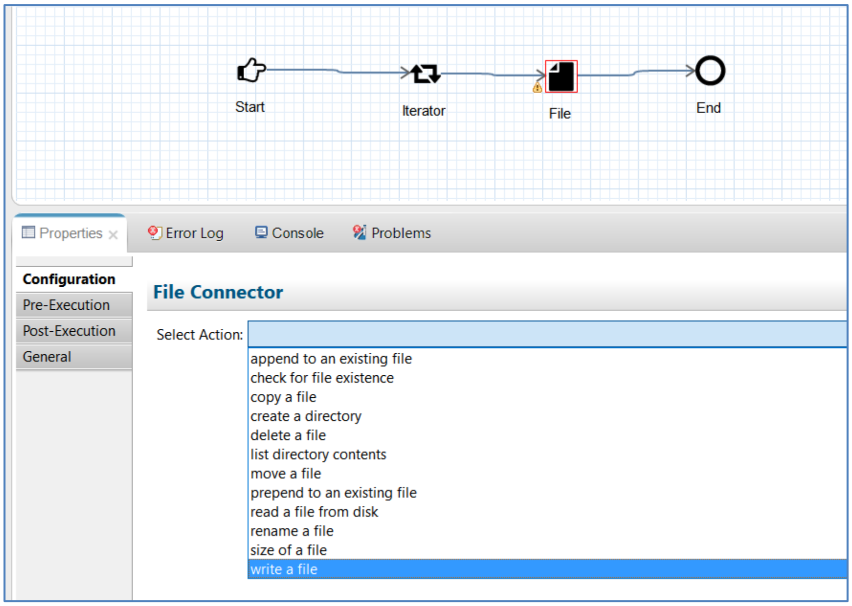
## Adding a File Object to Write a Record to the File System

1. From the Palette in the right pane, expand *Connectors* and drag the *File* object onto the line between the Iterator and End objects in your workspace area.

   The File object is added to your process flow, as shown in the following image.



2. From the Select Action drop-down list, select *write a file,* as shown in the following image.



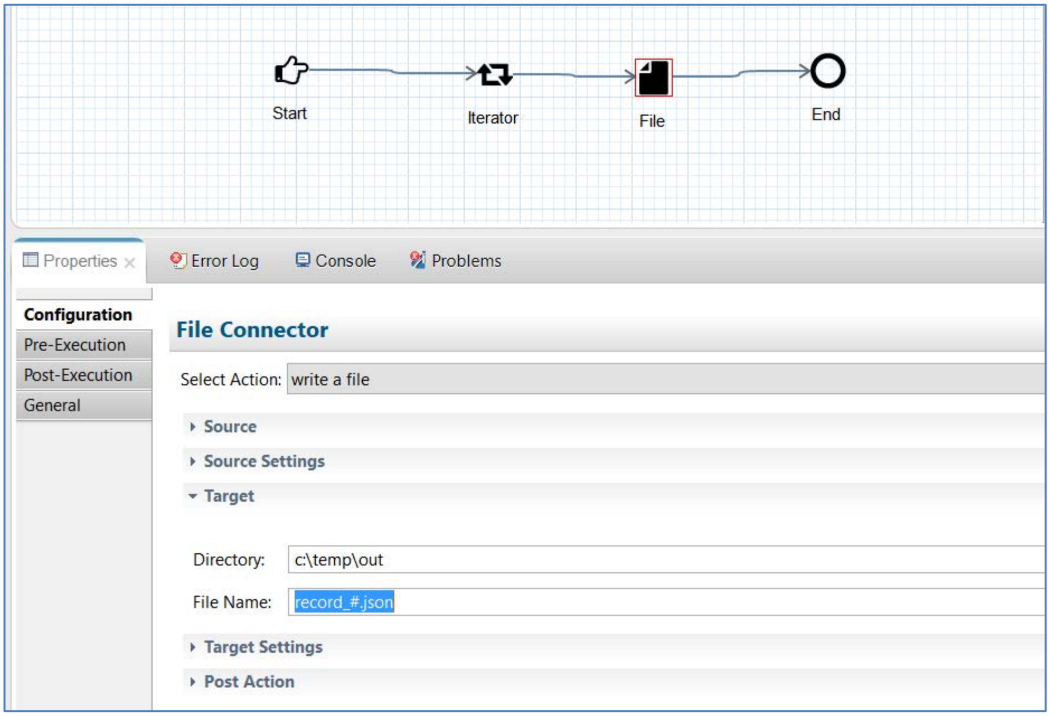
File Connector

Select Action: write a file

A list of configuration properties for this action is provided. The only two required parameters are for the definition of the output location and the file name to use.

3. Provide the following parameter values under the Target category.

| Parameter | Value |
|---|---|
| Directory | *c:\temp\out* |
| File Name | *record_#.json* |



Select Action: write a file

Directory: c:\temp\out

File Name: record_#.json

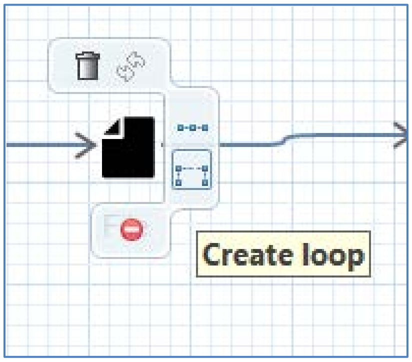
You are now ready to configure a loop from the File object, which writes single records to the file system, back to the Iterator.

## Configuring a Loop

1. Hover your mouse pointer over the File object.

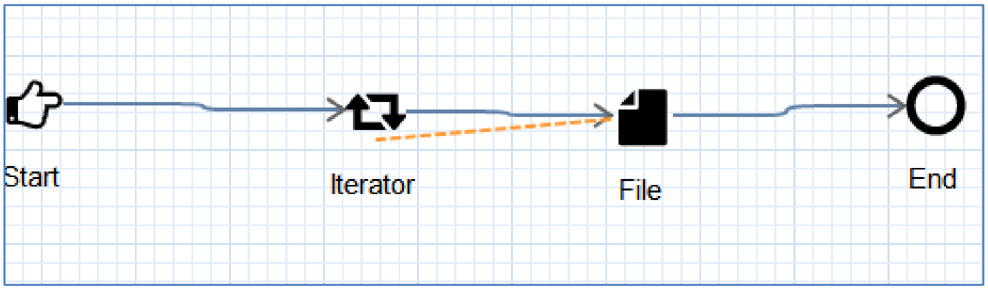
   A set of options is displayed.

2. Select and drag the *Create loop* operation to the Iterator object.



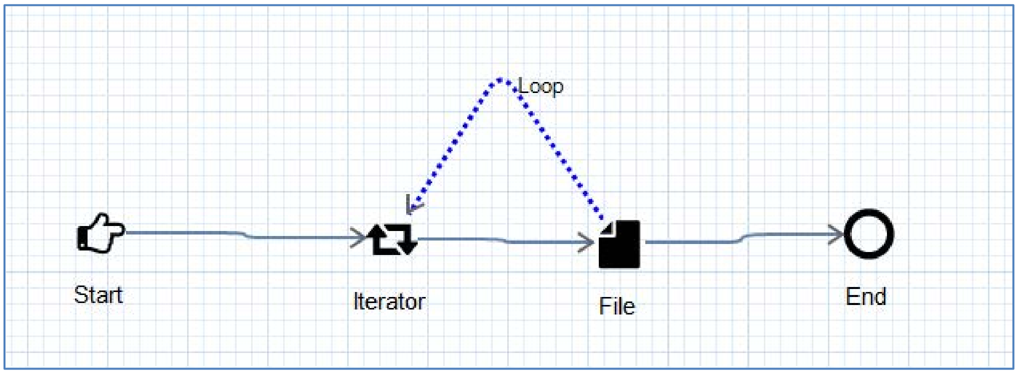
3. When a dashed orange line forms on the Iterator object, release the *Create loop* operation.

   The loop relationship will be created, as shown in the following image.



The default loop connector line is drawn to save space for complex process flows with many objects.

You can move the loop connector line accordingly for clarity or re-draw it, as shown in the following image.



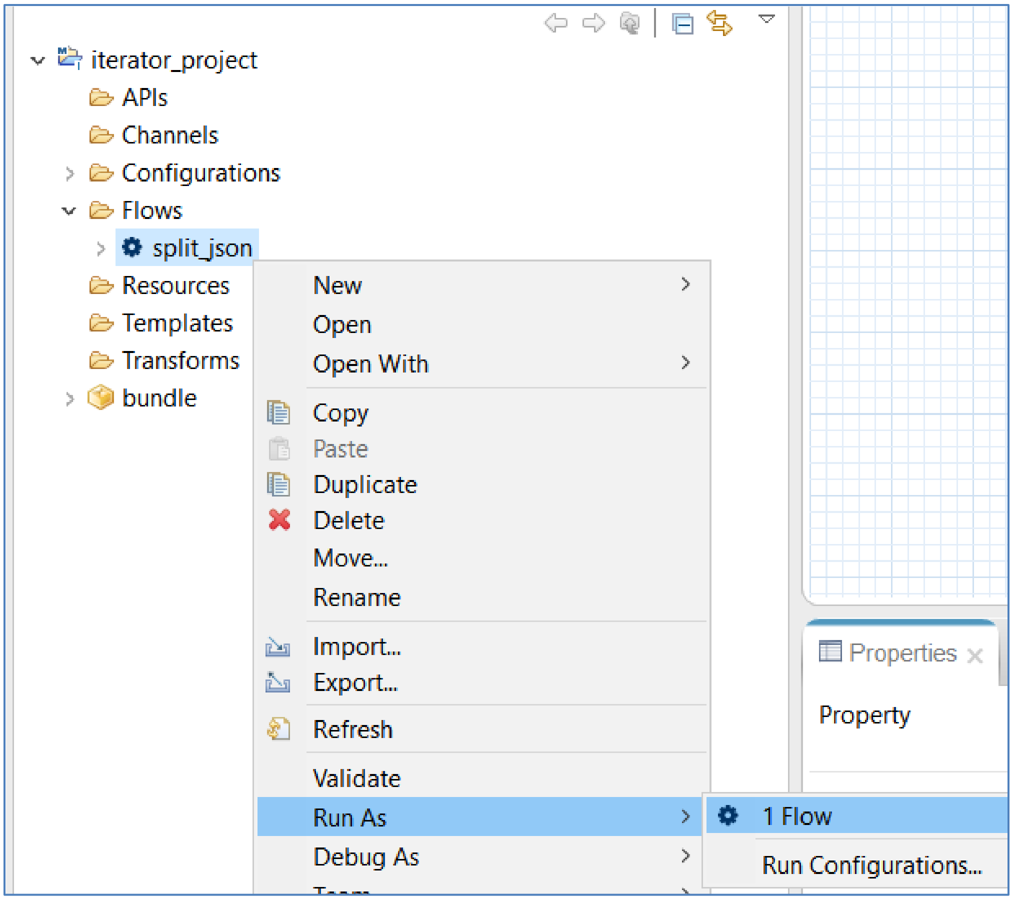
4. Save your work by clicking on the multi-disk icon 🖫 which saves all the components.

   You are now ready to test the process flow.

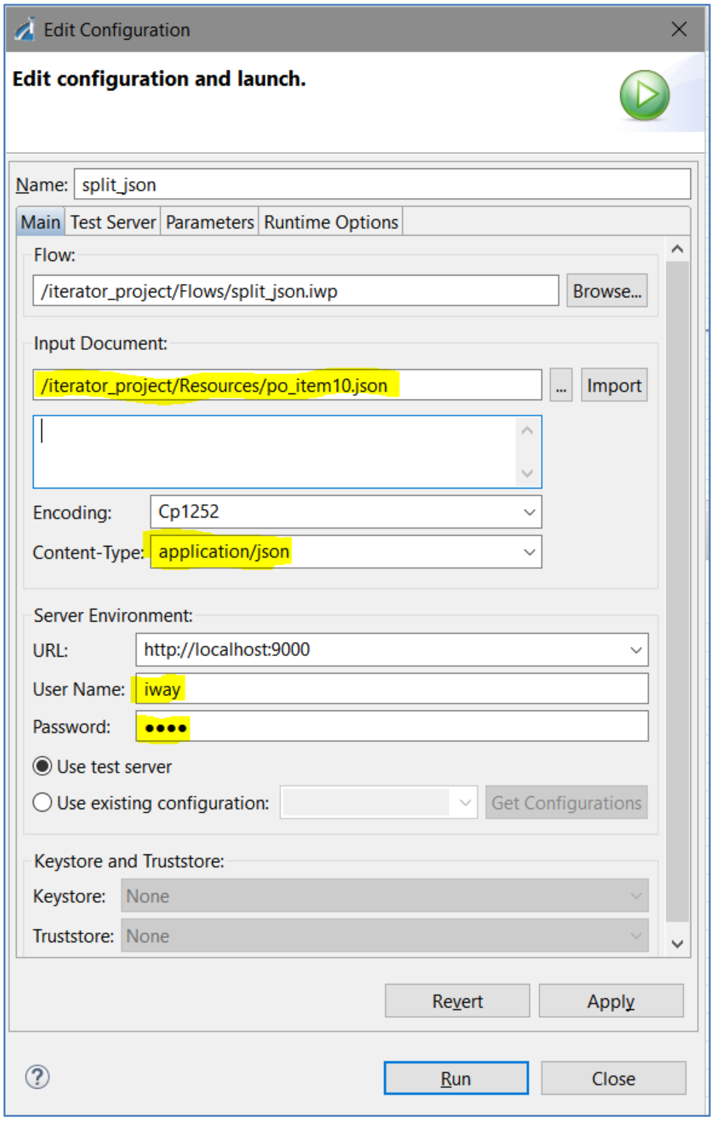
## Test Running Your Process Flow

This test run will simulate receiving a JSON batch document from an external source and processing this document. In an actual application, you would associate this process flow as the route of an inbound listener or call it as an API.

1. Right-click the *split_json* process flow under the project, select *Run As*, and then click *Flow* from the context menu, as shown in the following image.



The Edit configuration dialog opens, where you must specify values for the configuration parameters that will be used to test run the process flow.



Edit configuration and launch.

Name: split_json

Flow: /iterator_project/Flows/split_json.iwp

Input Document: /iterator_project/Resources/po_item10.json

Encoding: Cp1252

Content-Type: application/json

Server Environment:

URL: http://localhost:9000

User Name: iway

Password: ••••

● Use test server

○ Use existing configuration:

Keystore and Truststore:

Keystore: None

Truststore: None

2. Specify values for the parameters, as described in the following table.

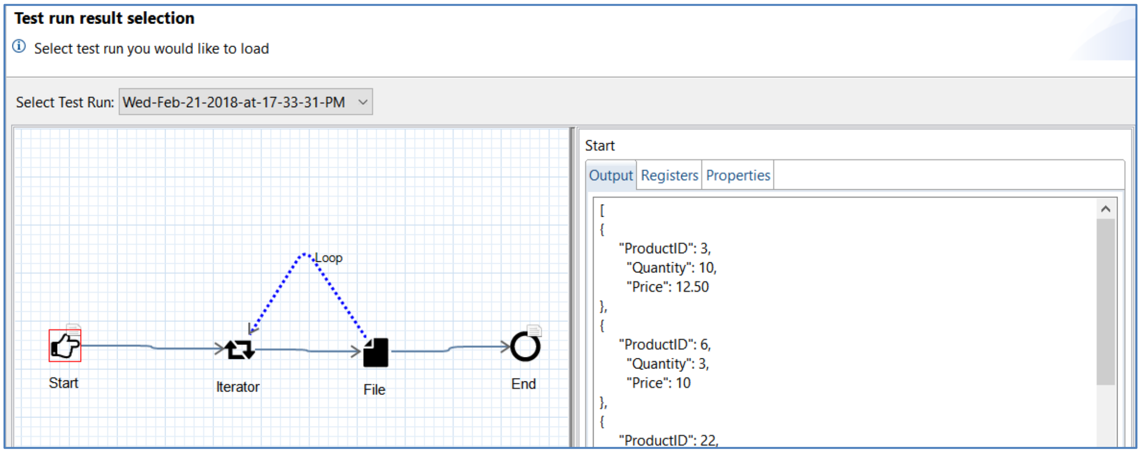
| Parameter | Value/Action |
|---|---|
| Input Document | Delete the default xml data that is displayed in the field.<br>Click *Import*, navigate to the location where the *po_item10.json* sample JSON document is located, and save it into the *Resources* folder when prompted. |
| Content-Type | Select *application/json* from the drop-down list. |
| User Name | *iway* (default user name) |
| Password | *iway* (default password) |

3. Click *Run*.

This action will create and start a test server where the process flow will be executed using the provided input document. You can also test against an existing application or provide definitions for a test server to be created.

The test run process will take about a minute, as a new server instance must be started for the test environment.

Upon successful startup, the *Test run result selection* dialog opens, as shown in the following image.



**Test run result selection**

ⓘ Select test run you would like to load

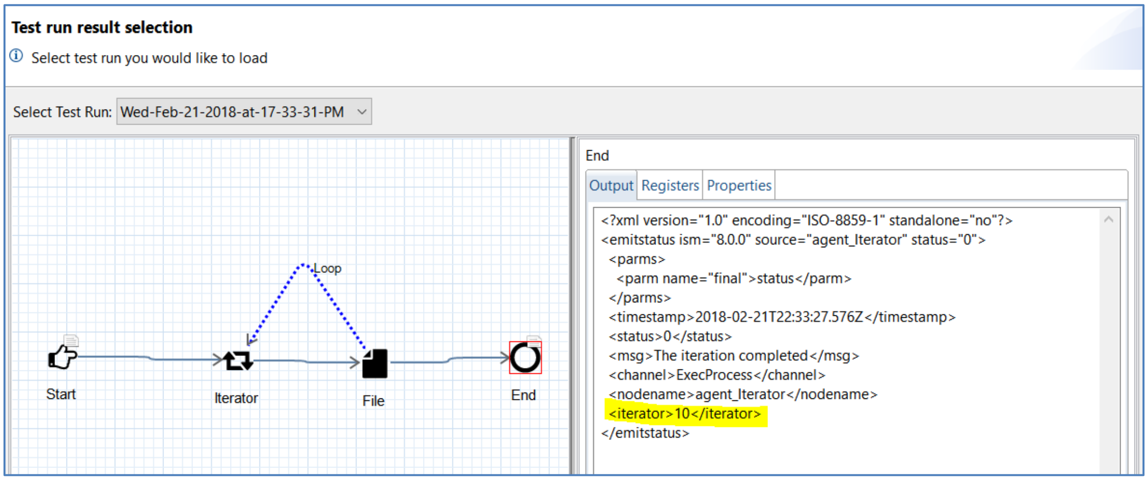
Select Test Run: Wed-Feb-21-2018-at-17-33-31-PM

Start

```
[
{
    "ProductID": 3,
      "Quantity": 10,
      "Price": 12.50
},
{
    "ProductID": 6,
      "Quantity": 3,
      "Price": 10
},
{
    "ProductID": 22,
```

Clicking the Start object in the process flow allows you to view the actual input document in the right pane.

Clicking the End object allows you to view the status response and the count of the iterators, which should be equal to 10, as shown in the following image.



**Test run result selection**

ⓘ Select test run you would like to load

Select Test Run: Wed-Feb-21-2018-at-17-33-31-PM

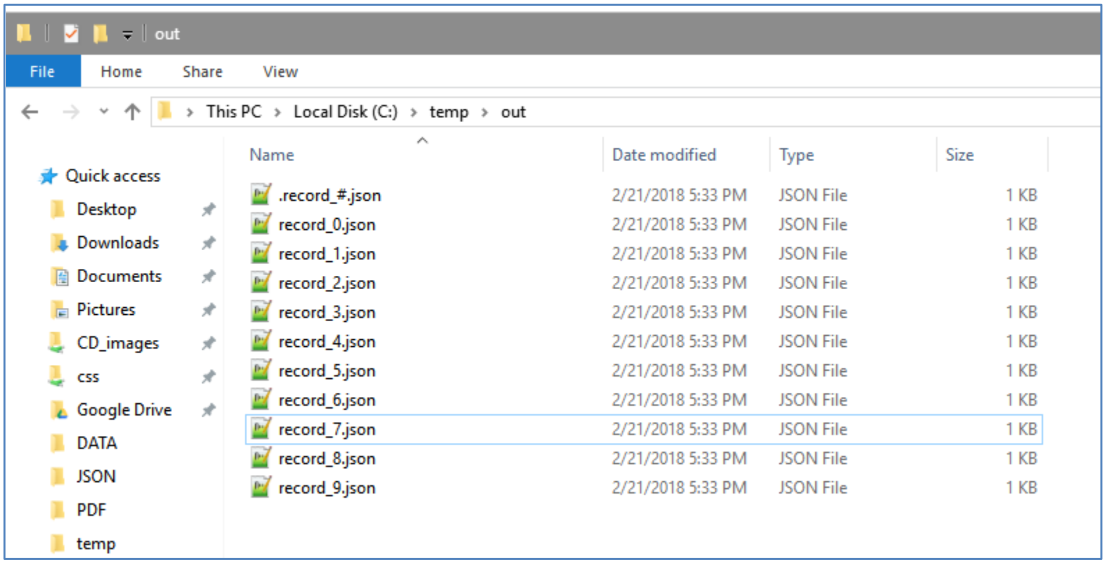
End

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<emitstatus ism="8.0.0" source="agent_Iterator" status="0">
  <parms>
    <parm name="final">status</parm>
  </parms>
  <timestamp>2018-02-21T22:33:27.576Z</timestamp>
  <status>0</status>
  <msg>The iteration completed</msg>
  <channel>ExecProcess</channel>
  <nodename>agent_Iterator</nodename>
  <iterator>10</iterator>
</emitstatus>
```

4. In your file system, navigate to the following directory:

```
C:\temp\out
```

This is the location where the 10 individual records (sub-documents) from the original input JSON batch document are now written, as shown in the following image.



| Name | Date modified | Type | Size |
|---|---|---|---|
| .record_#.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_0.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_1.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_2.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_3.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_4.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_5.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_6.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_7.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_8.json | 2/21/2018 5:33 PM | JSON File | 1 KB |
| record_9.json | 2/21/2018 5:33 PM | JSON File | 1 KB |

Notice that each file name has a counter, which was configured when *record_#.json* was specified as a value for the File Name parameter in the File object.  The first file, which has the # indicator, is a count keeper used for internal purposes.

You can open each individual record in a text editor or JSON editor to view its data.