

# Table of contents

---

- [Table of contents](#)
- [JSON-Repeat](#)
  - [Description](#)
  - [Compatibility](#)
  - [Getting Started](#)
  - [Output Connections](#)
  - [Flow Element](#)
    - [General Properties](#)
    - [Working Mode Depending Properties](#)
      - [JSON-Dataset](#)
  - [Examples](#)
  - [Error handling](#)
  - [Private data](#)

# JSON-Repeat

---

## Description

---

JSON-Repeat allows to inject new switch jobs for each element of a JSON array. The respective array element is appended as a dataset to the associated job.

It is also possible to define an "Ungroup" key, which can be used to combine all these jobs again afterwards.

## Compatibility

---

Switch Fall 2022 and higher.

## Getting Started

---

Use one of our sample flows and drop a sample file into the flow.

## Output Connections

---

This app requires one incoming connection - more incoming connections are allowed. The app supports traffic light outgoing connections of the following types:

- Log success: carries the created jobs
- Data error: carries the incoming job if the operation fails at the first attempt.
- Data success: carries the incoming job after the operation succeeds. If there are no data success connections the output is simply suppressed (with logging a warning).

## Flow Element

---

### General Properties

Property	Value	Description
<i>Working Mode</i>	enum [ JSON-File   JSON-Dataset ]	Defines the source of the input file
<i>JSON Path</i>	String	Defines a JSON Path expression to the array of the JSON-dataset; we recommend using <a href="https://jsonpath.com/">https://jsonpath.com/</a> to test your queries; checkout <a href="https://www.npmjs.com/package/jsonpath">https://www.npmjs.com/package/jsonpath</a> and <a href="https://www.npmjs.com/package/jsonpath-plus">https://www.npmjs.com/package/jsonpath-plus</a> for more information
<i>Ungroup Key</i>	String	Defines an <b>Ungroup</b> key, which can be used to combine all these jobs with the 'Assemble Job' element again afterwards.
<i>Dataset Name</i>	String	Defines the dataset name of the resulting JSON

### Working Mode Depending Properties

#### JSON-Dataset

Property	Value	Description
<i>Master Dataset Name</i>	String	Defines the name of the input dataset

## Examples

---

### Input

**Job-Name** : "sample.json"

```

{
  "Account": {
    "Account Name": "Firefly",
    "Order": [
      {
        "OrderID": "order103",
        "Product": [
          {
            "Product Name": "Bowler Hat",
            "ProductID": 858383,
            "Quantity": 2
          },
          {
            "Product Name": "Trilby hat",
            "ProductID": 858236
          }
        ]
      },
      {
        "OrderID": "order104",
        "Product": [
          {
            "Product Name": "Bowler Hat",
            "ProductID": 858383,
            "Quantity": 4
          },
          {
            "ProductID": 345664,
            "Product Name": "Cloak",
            "Quantity": 1
          }
        ]
      }
    ]
  }
}

```

## Configuration

JSON Path= \$.Account.Order

## Result

The JSON Path selects the order array and for each object a new job will be injected. The resulting jobs will be named like `<jobNameProper>_<index>.<ext>` and send to the log success connection.

Job 1: test\_0.pdf

Dataset:

```
{
  "OrderID": "order103",
  "Product": [
    {
      "Product Name": "Bowler Hat",
      "ProductID": 858383,
      "Quantity": 2
    },
    {
      "Product Name": "Trilby hat",
      "ProductID": 858236
    }
  ]
}
```

Job 2: test\_1.pdf

Dataset:

```
{
  "OrderID": "order104",
  "Product": [
    {
      "Product Name": "Bowler Hat",
      "ProductID": 858383,
      "Quantity": 4
    },
    {
      "ProductID": 345664,
      "Product Name": "Cloak",
      "Quantity": 1
    }
  ]
}
```

## Error handling

This app uses two types of errors:

- job data: if an handled error occurs (e.g. wrong file format), the error message is logged in the switch messages.
- job fail: if any other error occurs, job will fail and gets sent to the problem jobs folder. The thrown error gets logged as error and can be looked up in the switch messages.

## Private data

---

The following private data tags will always be set:

| - | - | | **Tag** | **Value** | **Type** | **Description** | json.repeat.index | Number | The index of the current element / file

The following private data tags will be set if an ungroup key is defined:

| - | - | - | | **Tag** | **Value** | **Type** | **Description** | <ungroupKey>.JobID | String | The unique name prefix of the incoming parent job  
<ungroupKey>.JobName | String | The name of the incoming parent job (without prefix)  
<ungroupKey>.NumFiles | Number | The length of the array that is iterated or in other words the total number of files injected in the flow for this parent job.  
<ungroupKey>.FileName | String | The name of this file as injected in the flow  
<ungroupKey>.FilePath | String | The relative path of this file within the parent job folder

The following private data tags will be set on error:

| - | - | - | | **Tag** | **Value** | **Type** | **Description** | lastErrorElement | String | the name of the flow element  
lastErrorId | jsonRepeatError | lastErrorCode | Number | an error code that defines the type of error that occurred  
lastErrorMessage | String | detailed error message

### Error Codes:

```
enum ERROR_CODES {  
    generalError = 0,  
    fileHandlingError = 1,  
    fileFormatError = 2,  
    conversionError = 3,  
    invalidParameterValue = 4,  
    parsingError = 5,  
}
```