Specifications

info This document describes configuration files with version 2.2. Configurations with version 1.x are deprecated, but can still be used with newer versions of Conduit. warning If you have a pipeline configuration file with version 1.x and multiple processors please update it to version 2.2 to ensure that processors are ordered correctly. The pipeline configuration file is a YAML file with a specific structure, that allows you to configure one or more pipelines. All pipelines, defined in the configuration file, will be provisioned in the order in which they are defined.

You can get a quick overview of the fields by looking at the following example:

example_pipeline.yml version :

2.2

Parser version

pipelines :

A list of pipeline configurations

-

id : pipeline1

Pipeline ID [required]

status : running

Pipeline status at startup (running or stopped)

name : pipeline1 - name

Pipeline name

description : desc

Pipeline description

connectors :

A list of connector configurations

-

id : con1

Connector ID [required]

type : source

Connector type (source or destination) [required]

plugin : builtin : file

Connector plugin [required]

name : my - file - source

Connector name

settings :

A map of configuration keys and values for the plugin (specific to the chosen plugin)

path : ./file1.txt

This property is specific to the file plugin

-

id : con2 type : destination plugin : builtin : file name : my - file - destination settings : path : ./file2.txt processors :

A list of processor configurations, processors are attached to the connector

-

id : proc1

Processor ID [required]

plugin : custom.javascript

Processor plugin name [required]

condition :

'{{ eq .Metadata.foo "bar" }}'

Condition (Go template expression) that dictates if the record will be passed to the processor or not.

workers :

2

Number of parallel workers

settings :

A map of configuration keys and values for the processor (specific to the chosen processor plugin)

script :

function process(record) { return record; // pass through } processors :

A list of processor configurations, processors are

attached to the pipeline

-

id : proc2

Processor ID [required]

plugin : field.set

Processor type [required]

settings :

A map of configuration keys and values for the processor (specific to the chosen processor plugin)

field : .Payload.After.key value :

{ ENV_VAR }

You can use environment variables by wrapping them in a dollar sign and curly braces {}

dead-letter-queue :

Dead-letter queue (DLQ) configuration

plugin :

"builtin:file"

DLQ Connector plugin

settings :

A map of configuration keys and values for the plugin (specific to the chosen plugin)

path :

"./dlq.out" window-size :

5

DLQ nack window size

window-nack-threshold :

2

DLQ nack window threshold

version

- Data Type
- : String
- Required
- : No
- Default
- : latest (2.2
-)
- Allowed Values
- :1.0
- ,1.1
- ,2.0
- ,2.1
- ,2.2
- Description
- : Version defines the schema version for the pipeline
- configuration file and controls which parser is used to decode it.

pipelines

- Data Type
- : Sequence node containingpipeline
- nodes
- Required
- : Yes
- Default
- : None
- Description
- : The node contains a list ofpipeline
- definitions. Pipelines will be provisioned in the order in which they are
- defined.

pipeline

- Data Type
- : Mapping node
- Required
- : n/a
- Default
- : None
- Description
- : This node contains the configuration of a single pipeline.

id

- Data Type
- : String
- Required
- : Yes
- Default
- : None
- Description
- : ID of thepipeline
- . It should uniquely
- identify the pipeline across all pipelines. If multiple pipelines use the same
- ID, none of them will be provisioned.

Warning : Changing this property will cause the pipeline to be recreated and start from the beginning.

status

- Data Type
- : String
- Required
- : No
- Default
- :stopped
- Allowed Values

- :stopped
- ,running
- Description
- : This field controls the status of the pipeline at startup. If
- set torunning
- the pipeline will be automatically started, if set tostopped
- the pipeline will be provisioned without being started.

name

- Data Type
- : String
- Required
- : No
- Default
- : Same as pipelineid
- Description
- : Human readable name for the pipeline. Needs to be unique
- across all pipelines (it is used as a label in pipelinemetrics
-).

description

- Data Type
- : String
- Required
- : No
- Default
- : None
- Description
- : Human readable description of the pipeline.

connectors

- Data Type
- : Sequence node containingconnector
- nodes
- Required
- : No
- Default
- : None
- Description
- : The node contains a list ofconnector
- definitions. The order of the connectors has no effect on the provisioned
- pipeline.

processors

- Data Type
- : Sequence node containingprocessor
- nodes
- Required
- : No
- Default
- : None
- Description
- : The node contains a list ofprocessor
- definitions. These processors process all records flowing through the
- pipeline. They are executed after source processors and before destination
- processors. The processors are executed in the order in which they are defined
- in the list.

dead-letter-queue

- Data Type
- : Mapping node
- Required
- : No

- Default
- : See sub-fields
- Description
- : This node contains the dead-letter queue configuration. Read
- more aboutdead-letter queues
- in Conduit.

plugin

- Data Type
- : String
- Required
- : No
- Default
- :builtin:log
- Description
- : This node references the destination connector plugin used
- for storing dead-letters. See how toreference a connector
- .

settings

- Data Type
- : Mapping node
- Required
- : No
- Default
- :{"level": "warn", "message": "record delivery failed"}
- Description
- : A map of configuration keys and values for the dead-letter
- queue connector. These settings depend on the value of the dead-letter queueplugin
- . Check the chosen destination connector's documentation
- for a list of settings it supports and requires.

window-size

- Data Type
- : Integer
- Required
- : No
- Default
- :1
- Description
- : Defines the nack window size. Seedead-letter queue
- .

window-nack-threshold

- Data Type
- : Integer
- Required
- : No
- Default
- :0
- Description
- : Defines the nack window threshold. Seedead-letter queue
- .

connector

- Data Type
- : Mapping node
- Required
- : n/a
- Default
- : None
- Description

- : This node contains the configuration of a single connector.

id

- Data Type
- : String
- Required
- : Yes
- Default
- : None
- Description
- : ID of theconnector
- . It should uniquely
- identify the connector inside the pipeline. If multiple connectors inside the
- pipeline use the same ID, provisioning will fail.

Warning : Changing this property will cause the connector to be recreated and start from the beginning.

type

- Data Type
- : String
- Required
- : Yes
- Default
- : None
- Allowed Values
- :source
- ,destination
- Description
- : Defines if the connector is a source or a destination. A
- valid pipeline needs at least one source and one destination.

Warning : Changing this property will cause the connector to be recreated and start from the beginning.

plugin

- Data Type
- : String
- Required
- : Yes
- Default
- : None
- Description
- : This node references the connector plugin. See how toreference a connector
- .

Warning : Changing this property will cause the connector to start from the beginning.

name

- Data Type
- : String
- Required
- : No
- Default
- : Same as connectorid
- Description
- : Human readable name for the connector.

settings

- Data Type
- : Mapping node
- Required
- : Yes
- Default
- : None

- Description
- : A map of configuration keys and values for the connector
- plugin. These settings depend on the value of connectortype
- andplugin
- . Check the chosen connector's documentation for a list
- of settings it supports and requires.

processors

- Data Type
- : Sequence node containingprocessor
- nodes
- Required
- : No
- Default
- : None
- Description
- : The node contains a list ofprocessor
- definitions. These processors only process records coming from or flowing to a
- source or destination. The processors are executed in the order in which they
- are defined in the list.

processor

- Data Type
- : Mapping node
- Required
- : n/a
- Default
- : None
- Description
- : This node contains the configuration of a single processor.

id

- Data Type
- : String
- Required
- : Yes
- Default
- : None
- Description
- : ID of theprocessor
- . It should uniquely
- identify the processor inside the pipeline. If multiple processors inside the
- pipeline use the same ID, provisioning will fail.

plugin

- Data Type
- : String
- Required
- : Yes
- Default
- : None
- Description
- : Defines the processor's plugin name (e.g.field.set
-). Check out theprocessors documentation
- to find the list of builtin processors
- we provide.

condition

- Data Type
- : String
- Required
- : No

- Default
- :true
- (all records will be passed to the processor)
- Description
- : AGo template expression
- that will be used as a condition to pass the record to the processor or not.
- The Go template expression will be evaluated using each record and needs to producetrue
- orfalse
- . If it producestrue
- the record will be passed to the processor,false
- means it will continue down the pipeline without getting processed.

settings

- Data Type
- : Mapping node
- Required
- : Yes
- Default
- : None
- Description
- : A map of configuration keys and values for the processor. The
- values in this map depend on the chosen processoplugin
- .

workers

- Data Type
- : Integer
- Required
- : No
- Default
- :1
- Description
- : Defines the number of workers that should execute this
- processor in parallel. The number needs to be larger than 0.Edit this page Previous Provisioning Next Getting Started