My steps to learn about Apache NiFi

Paulo Jerônimo, 2018-05-24 05:36:18 WEST

Table of Contents

Introduction

Recently I had work to produce a document with a comparison between two tools for Cloud Data Flow. I didn’t have any knowledge of this kind of technology before creating this document. Apache NiFi is one of the tools in my comparison document. So, here I describe some of my procedures to learn about it and take my own preliminary conclusions. I followed many steps on my own desktop (a MacBook Pro computer) to accomplish this task. This document shows you what I did.

Basically, to learn about Apache NiFi in order to do a comparison with other tool:

- I saw some videos about it.
- I wrote my own labs:
  - Running Apache NiFi inside a Docker container
  - Running Apache NiFi locally
  - Building a simple Data Flow *(under contruction)*
  - Developing a custom processor *(under contruction)*
  - Deploying Apache NiFi under Pivotal Cloud Foundry *(under contruction)*

About this document

This document was written using Vim (my favorite text editor) and its source code is in AsciiDoc format. The generated output formats (HTML and PDF) are build (and published in GitHub Pages) with Gradle. Read more about the generation processes of this document in README.adoc.

See the online version of this document in HTML format.

About me

You can read more about me on my cv.

Videos with a technical background

Prior to starting my own labs, I saw some introductory videos (available on YouTube):

- "Matt Burgess discusses Open Source Software & Apache nifi" (yA)
- "Spring with ApacheNiFi" (yB)
- "Hortonworks DataFlow powered by Apache NiFi" (yC)
- "How to navigate and build a dataflow in Apache Nifi" (yD)

Lab 1: Running Apache NiFi inside a Docker container

For me, the best way to start learning a new technology is by running all the stuff related to them inside a Docker container. By this way, I can abstract myself about the related installation procedures and go directly to the point.

So, In this tutorial, I present the steps to work with Apache NiFi using Docker.

Prerequisites

1. Docker installed.

Start/Restart

First start:

```
docker run --name nifi -p 9090:9090 -d -e NIFI_WEB_HTTP_PORT='9090' apache/nifi:latest
```

Restart (if was started any time before with the command below and stopped):

```
docker start nifi
```

Access to the UI

Open http://localhost:9090/nifi

Status

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED
STATUS              PORTS                                                  NAMES
3a506cfec5ab        apache/nifi:latest   "/bin/sh -c ${NIFI_B…"   10 hours ago
Up 10 hours         8080/tcp, 8443/tcp, 10000/tcp, 0.0.0.0:9090->9090/tcp   nifi
```

Stop

```
docker stop nifi
```

```
$ docker ps -a
CONTAINER ID        IMAGE                COMMAND                  CREATED
STATUS                        PORTS               NAMES
3a506cfec5ab        apache/nifi:latest   "/bin/sh -c ${NIFI_B…"   10 hours ago
Exited (137) 33 seconds ago                       nifi
```

Lab 2: Running Apache NiFi locally

Prerequisites

1. Java installed.

Installation

```
$ brew install nifi
....
######################################################################## 100.0%
    /usr/local/Cellar/nifi/1.6.0: 386 files, 1.2GB, built in 45 minutes 44 seconds
```

Start

```
$ nifi
Usage nifi {start|stop|run|restart|status|dump|install}
```

```
$ nifi start

Java home: /Users/pj/.sdkman/candidates/java/current
NiFi home: /usr/local/Cellar/nifi/1.6.0/libexec

Bootstrap Config File: /usr/local/Cellar/nifi/1.6.0/libexec/conf/bootstrap.conf
```

Access to the UI

Open http://localhost:8080/nifi/

Status

```
$ nifi status

Java home: /Users/pj/.sdkman/candidates/java/current
NiFi home: /usr/local/Cellar/nifi/1.6.0/libexec

Bootstrap Config File: /usr/local/Cellar/nifi/1.6.0/libexec/conf/bootstrap.conf

2018-04-29 08:02:13,153 INFO [main] org.apache.nifi.bootstrap.Command Apache NiFi is
currently running, listening to Bootstrap on port 58129, PID=4024
```

Stop

```
$ nifi stop

Java home: /Users/pj/.sdkman/candidates/java/current
NiFi home: /usr/local/Cellar/nifi/1.6.0/libexec

Bootstrap Config File: /usr/local/Cellar/nifi/1.6.0/libexec/conf/bootstrap.conf

2018-04-29 08:11:41,562 INFO [main] org.apache.nifi.bootstrap.Command Apache NiFi has
accepted the Shutdown Command and is shutting down now
2018-04-29 08:11:41,587 INFO [main] org.apache.nifi.bootstrap.Command Waiting for
Apache NiFi to finish shutting down...
2018-04-29 08:11:43,597 INFO [main] org.apache.nifi.bootstrap.Command NiFi has
finished shutting down.
```

Lab 3: Building a simple Data Flow

Prerequisites

1. Docker installed.

Step 1 - Create a Nifi docker container with default parameters

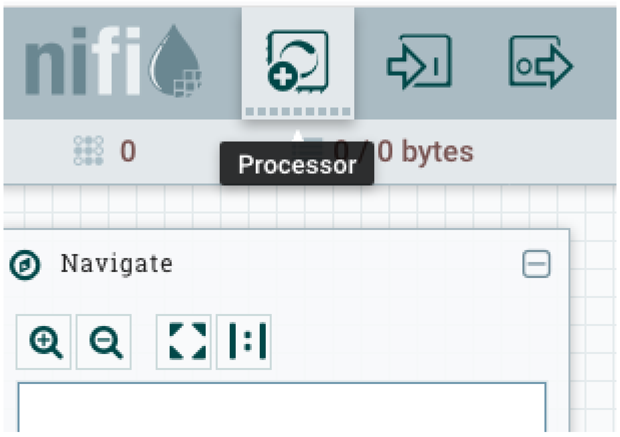
```
$ docker run --name nifi -p 8080:8080 -d apache/nifi
```

Step 2 - Access the UI and create two processors

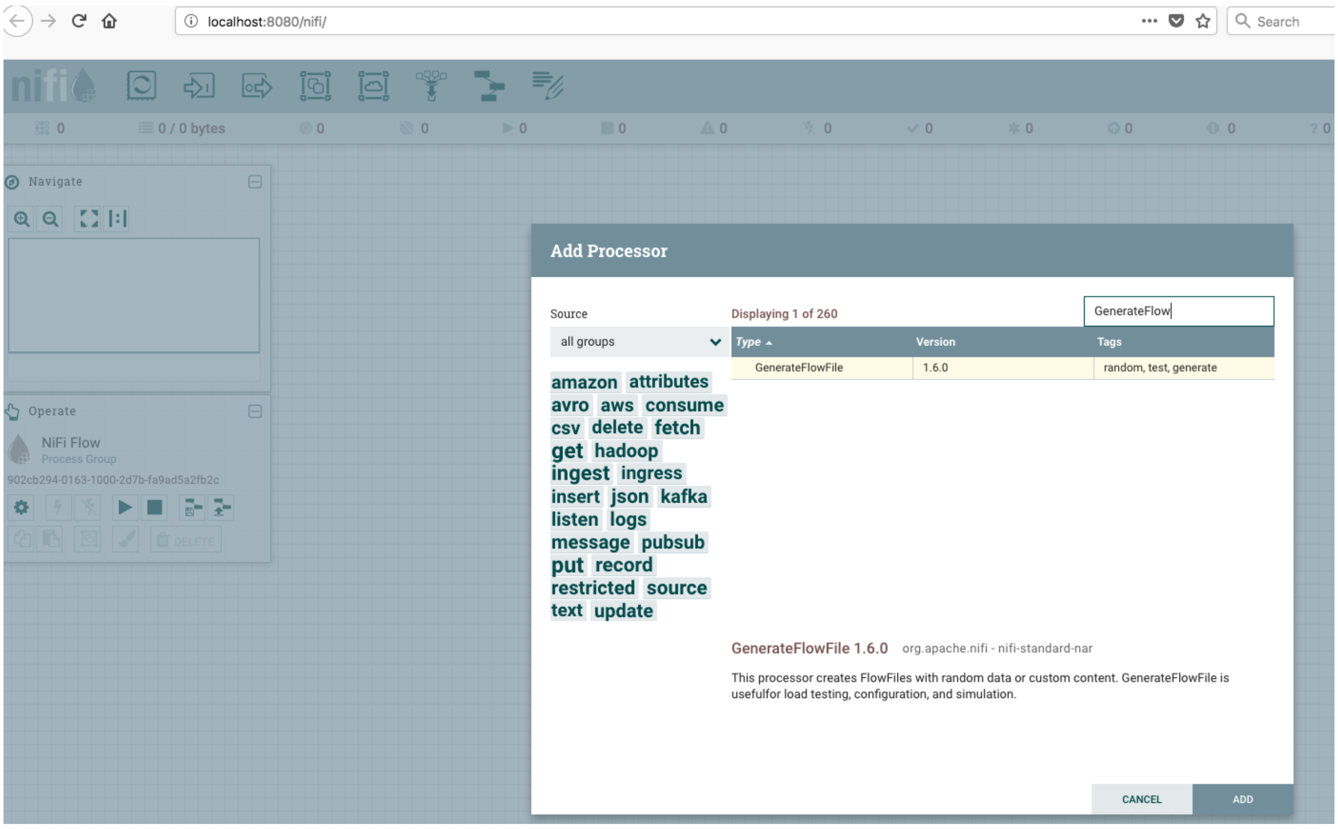
Open http://localhost:8080/nifi

Step 3 - Add and configure processor 1 (GenerateFlowFile)

Drag and drop a processor into canvas:



Search for a processor named `GenerateFlowFile`:


localhost:8080/nifi/
Navigate
Operate
NiFi Flow
Process Group
902cb294-0163-1000-2d7b-fa9ad5a2fb2c
DELETE
Add Processor
Source
all groups
amazon attributes avro aws consume csv delete fetch get hadoop ingest ingress insert json kafka listen logs message pubsub put record restricted source text update
CANCEL
ADD

Displaying 1 of 260

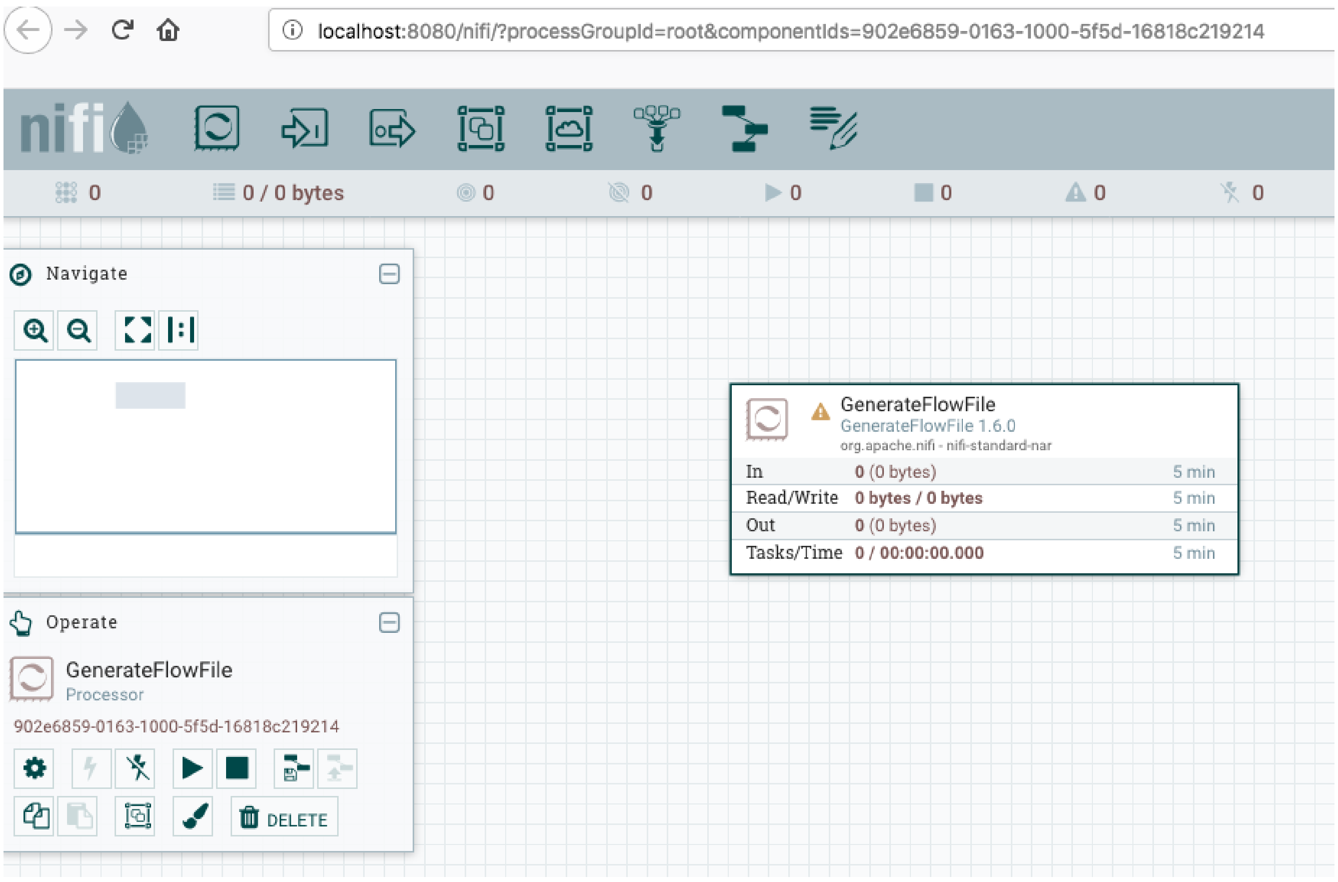
GenerateFlow

Type	Version	Tags
GenerateFlowFile	1.6.0	random, test, generate

GenerateFlowFile 1.6.0 org.apache.nifi - nifi-standard-nar

This processor creates FlowFiles with random data or custom content. GenerateFlowFile is usefulfor load testing, configuration, and simulation.

Click on Add and the processor will be added to the canvas:

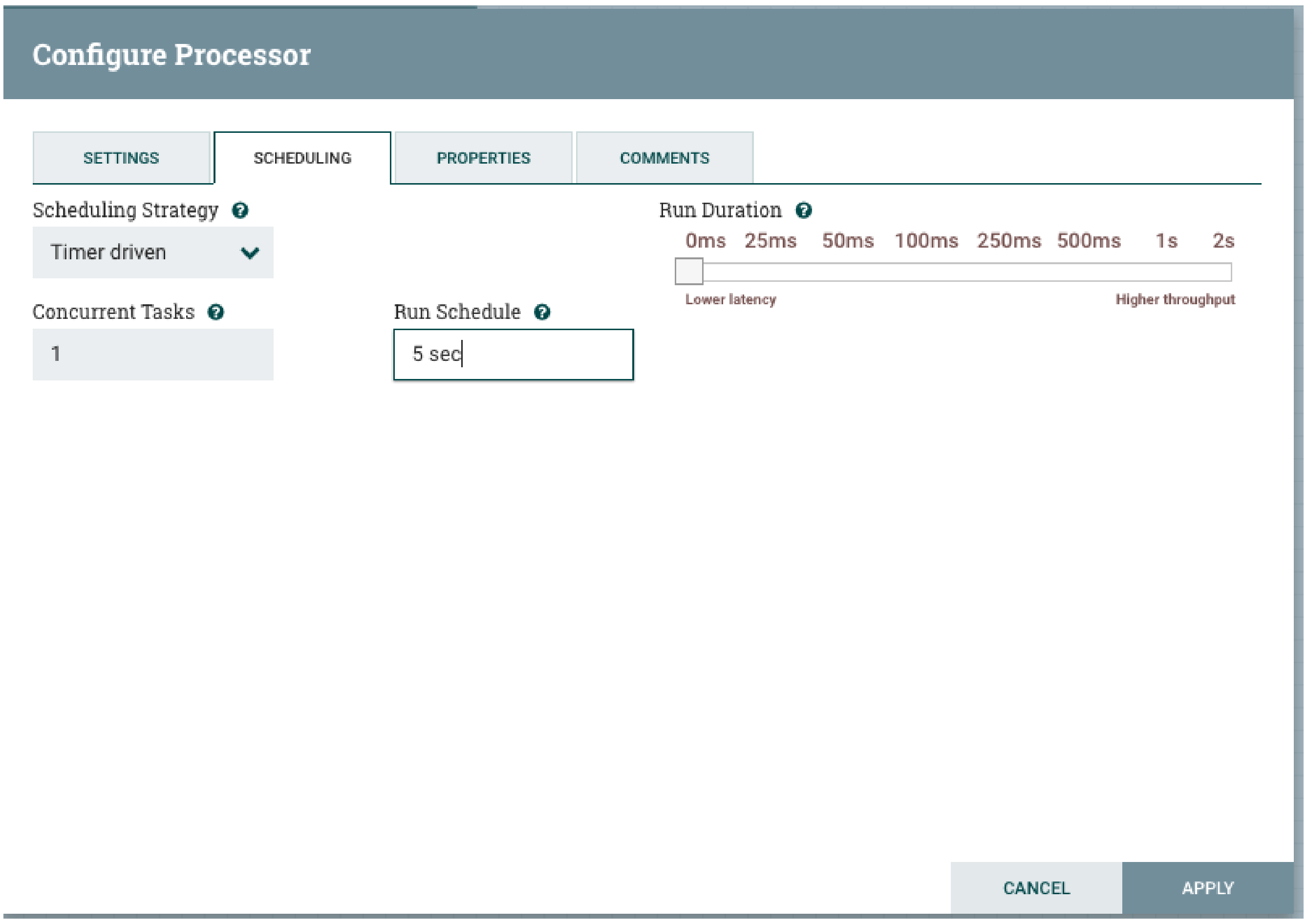

localhost:8080/nifi/?processGroupId=root&componentIds=902e6859-0163-1000-5f5d-16818c219214
Navigate
Operate
GenerateFlowFile
Processor
902e6859-0163-1000-5f5d-16818c219214
DELETE

GenerateFlowFile
GenerateFlowFile 1.6.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Configure the processor (2 steps):

Step 1 - Adjust the `Run Schedule` to `5 sec`:



Configure Processor

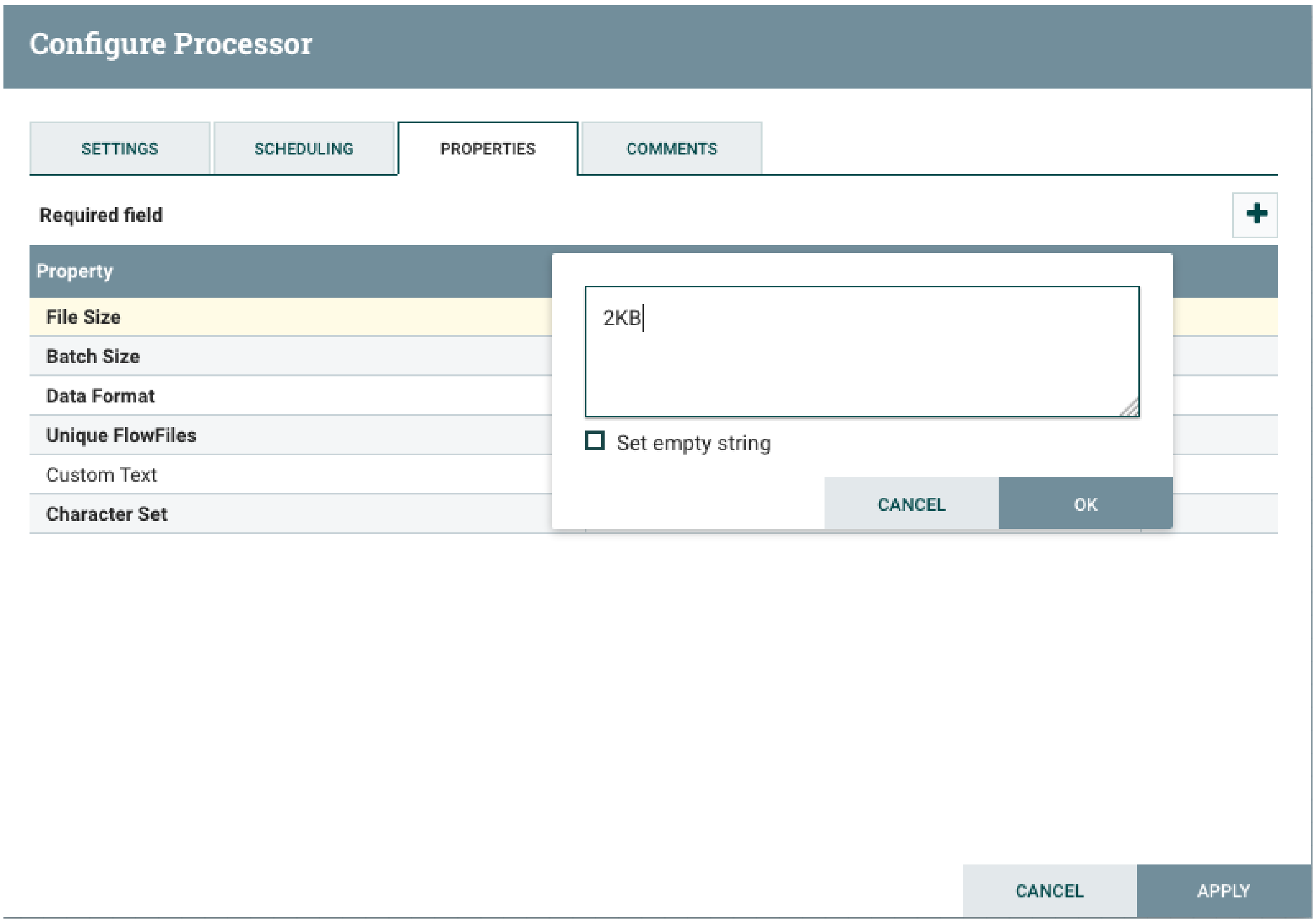
Scheduling Strategy: Timer driven

Concurrent Tasks: 1

Run Schedule: 5 sec

Run Duration: 0ms 25ms 50ms 100ms 250ms 500ms 1s 2s

Step 2 - Adjust the propertie `File Size` to `2KB`:



Configure Processor

Required field

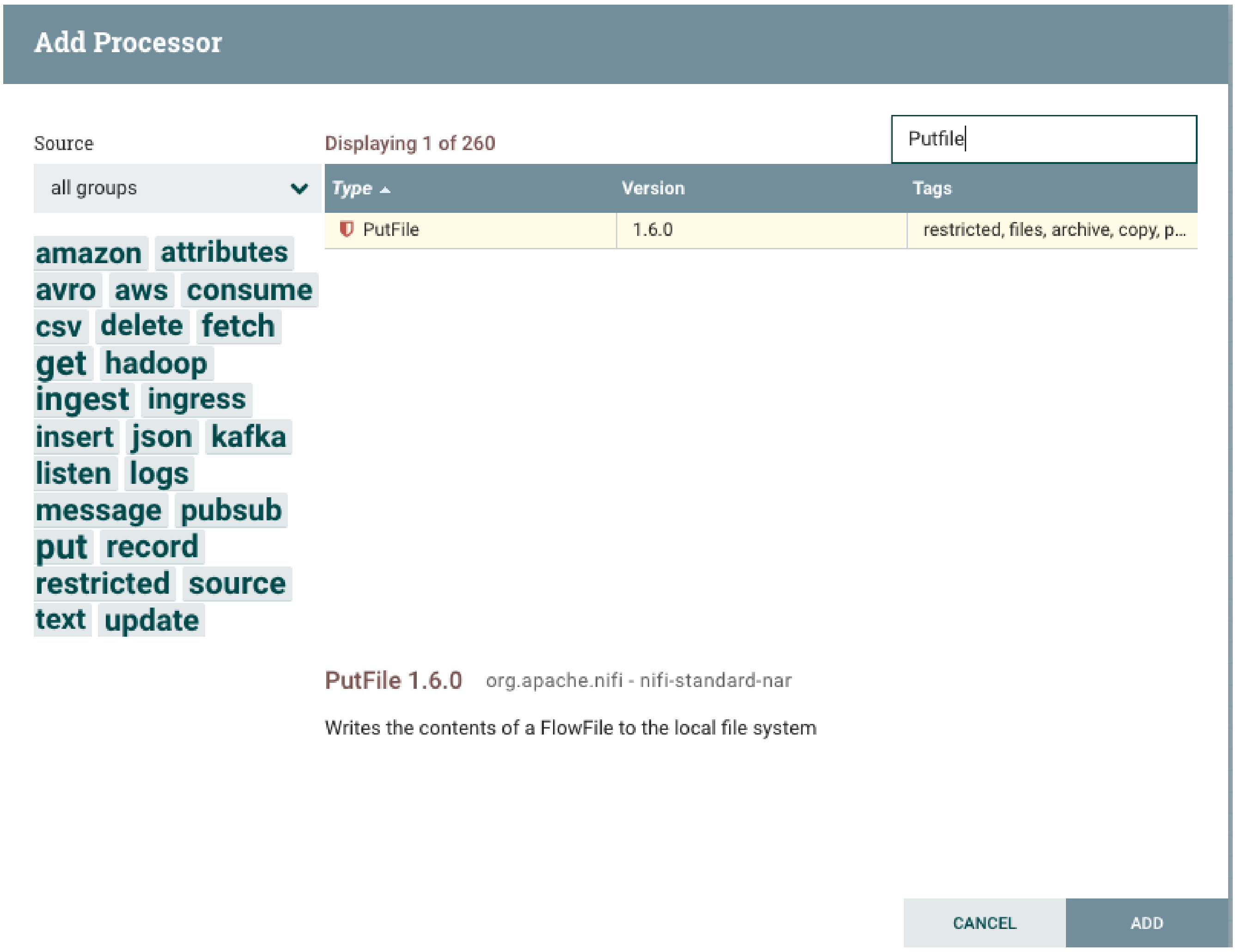
Property
File Size
Batch Size
Data Format
Unique FlowFiles
Custom Text
Character Set

2KB

☐ Set empty string

Step 4 - Add and configure processor 2 (Putfile)

Drag another processor into canvas. Search for `PutFile`:

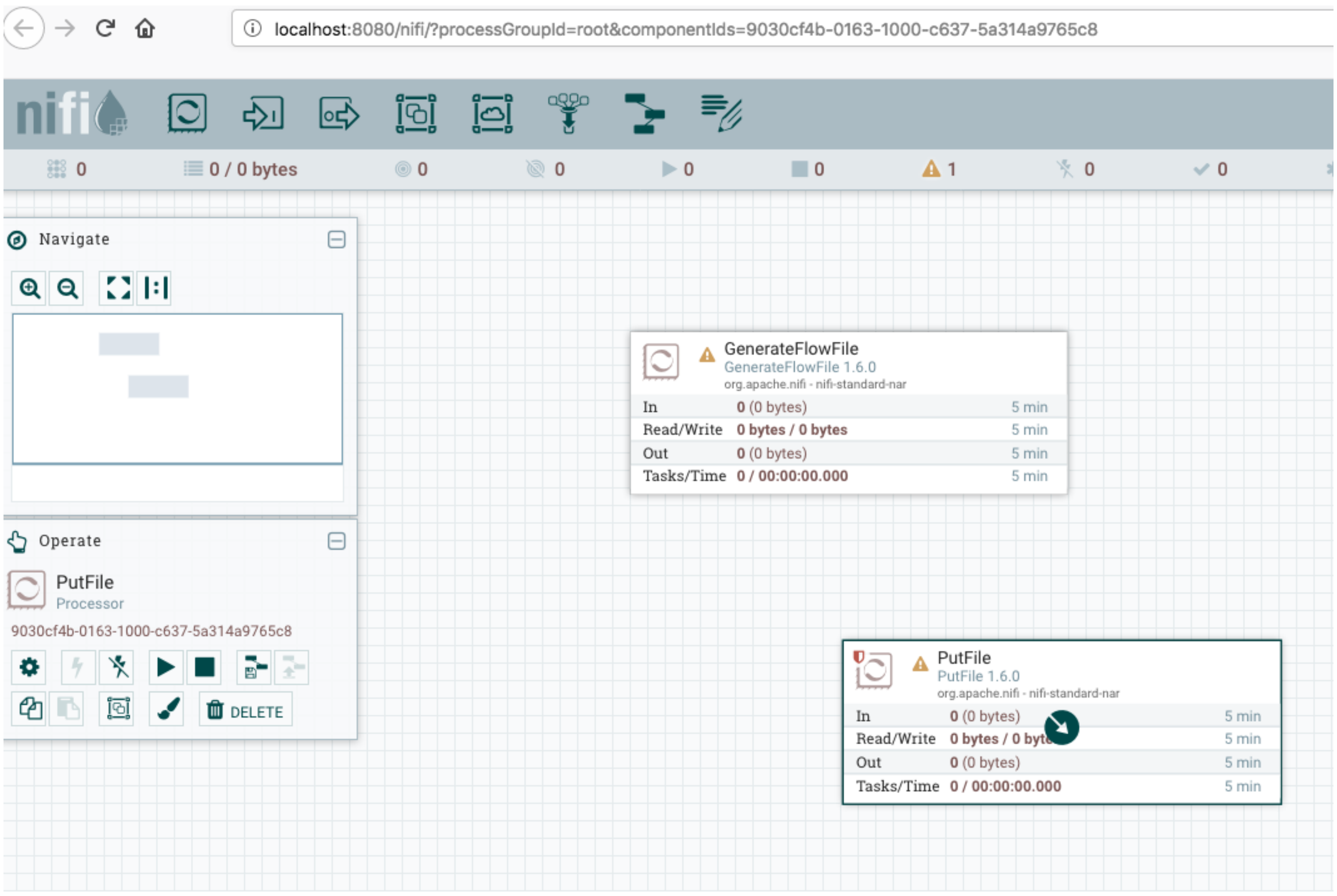

Add Processor
Source
all groups
amazon attributes avro aws consume csv delete fetch get hadoop ingest ingress insert json kafka listen logs message pubsub put record restricted source text update
Displaying 1 of 260
Putfile
CANCEL
ADD

Type	Version	Tags
PutFile	1.6.0	restricted, files, archive, copy, p...

PutFile 1.6.0 org.apache.nifi - nifi-standard-nar

Writes the contents of a FlowFile to the local file system

Add it to the canvas:


localhost:8080/nifi/?processGroupId=root&componentIds=9030cf4b-0163-1000-c637-5a314a9765c8
0
0 / 0 bytes
0
0
0
0
1
0
0
Navigate
Operate
PutFile
Processor
9030cf4b-0163-1000-c637-5a314a9765c8
DELETE

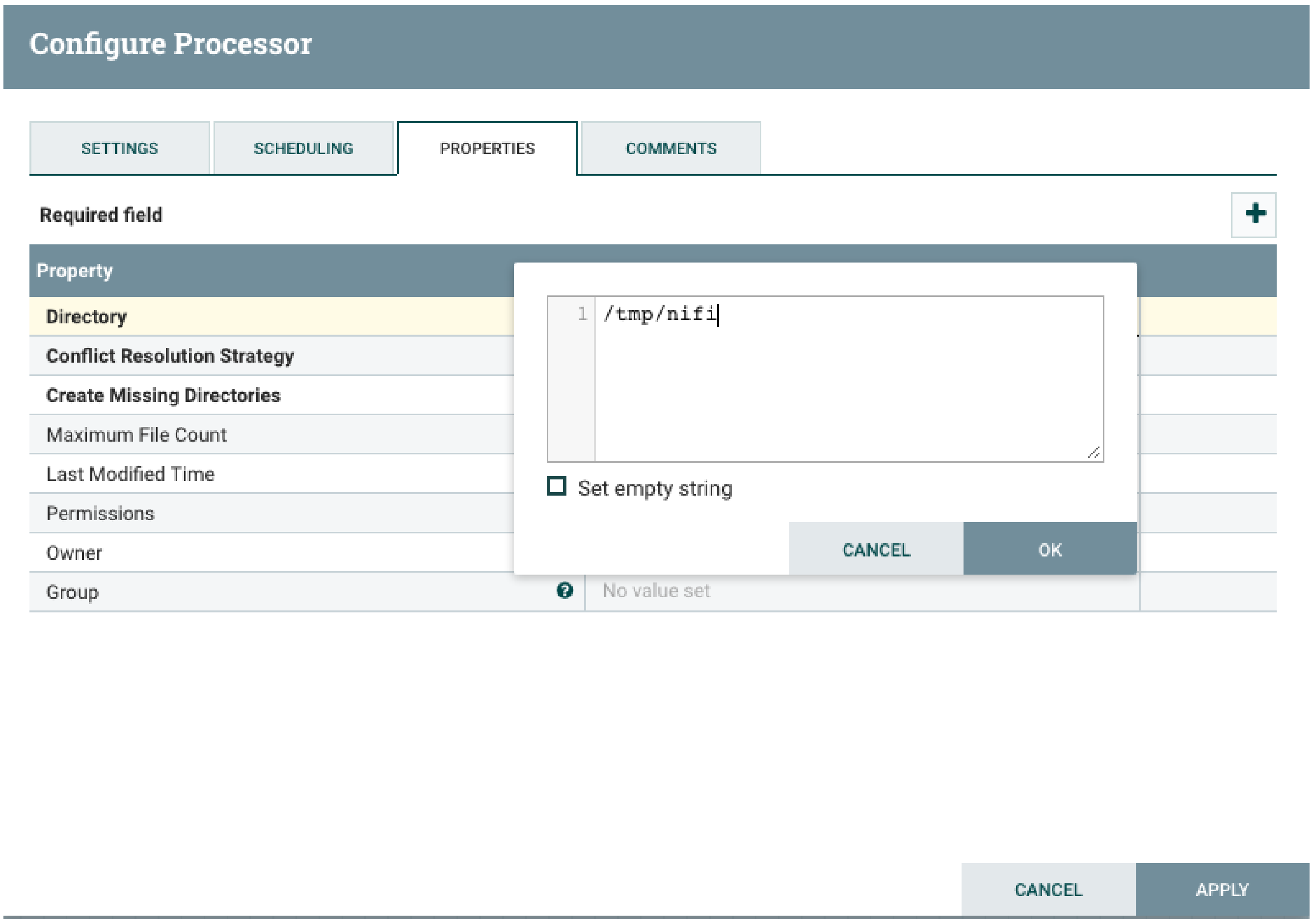
GenerateFlowFile
GenerateFlowFile 1.6.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

PutFile
PutFile 1.6.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Configure the `Directory` property to `/tmp/nifi`.



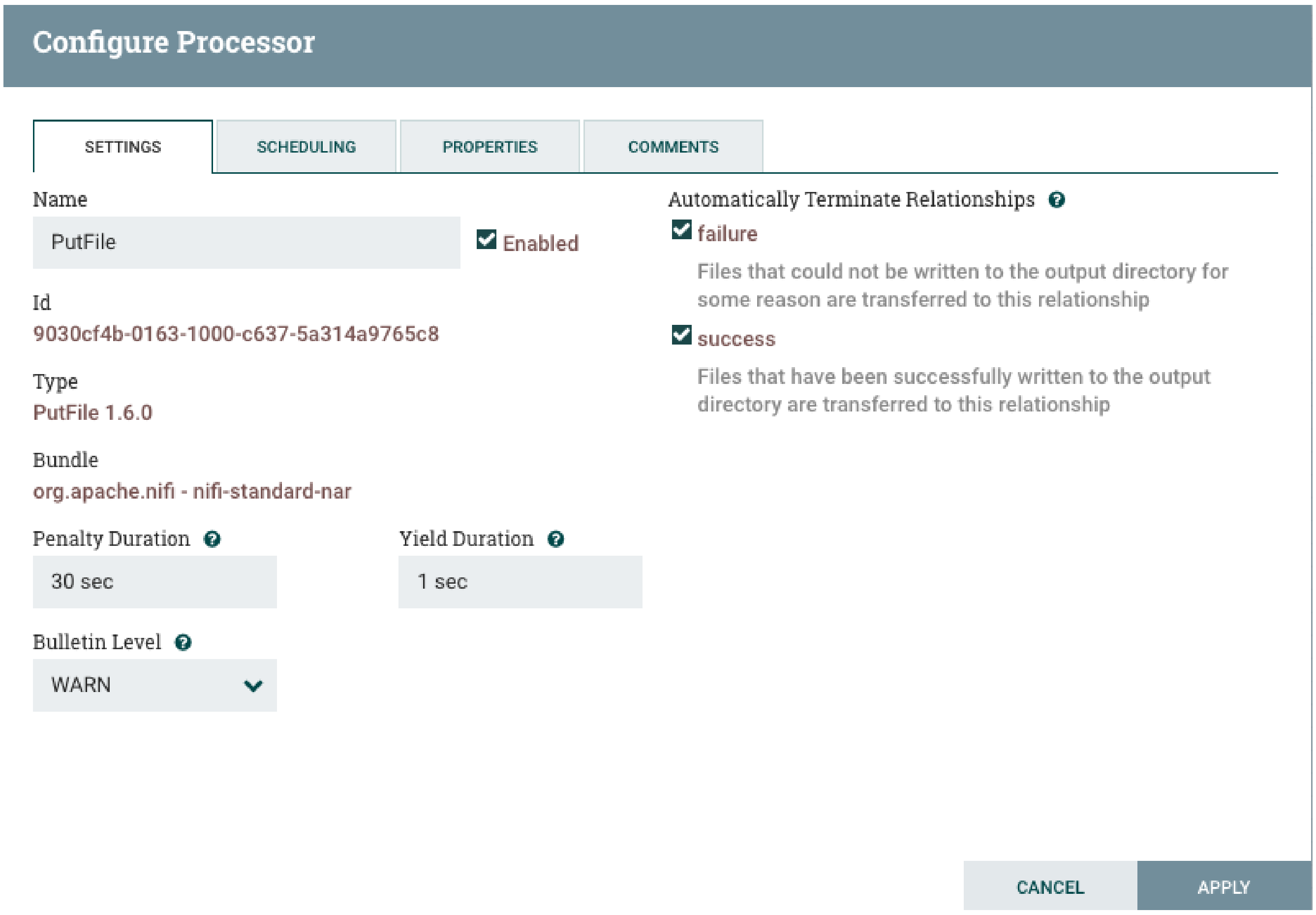
Required field

Property	
Directory	
Conflict Resolution Strategy	
Create Missing Directories	
Maximum File Count	
Last Modified Time	
Permissions	
Owner	
Group	No value set

```
/tmp/nifi
```

☐ Set empty string

Configure `Automatically Terminate RelationShips` by checking the boxes `failure` and `success`.


Configure Processor

Name
PutFile
☑ Enabled

Id
9030cf4b-0163-1000-c637-5a314a9765c8

Type
PutFile 1.6.0

Bundle
org.apache.nifi - nifi-standard-nar

Penalty Duration
30 sec

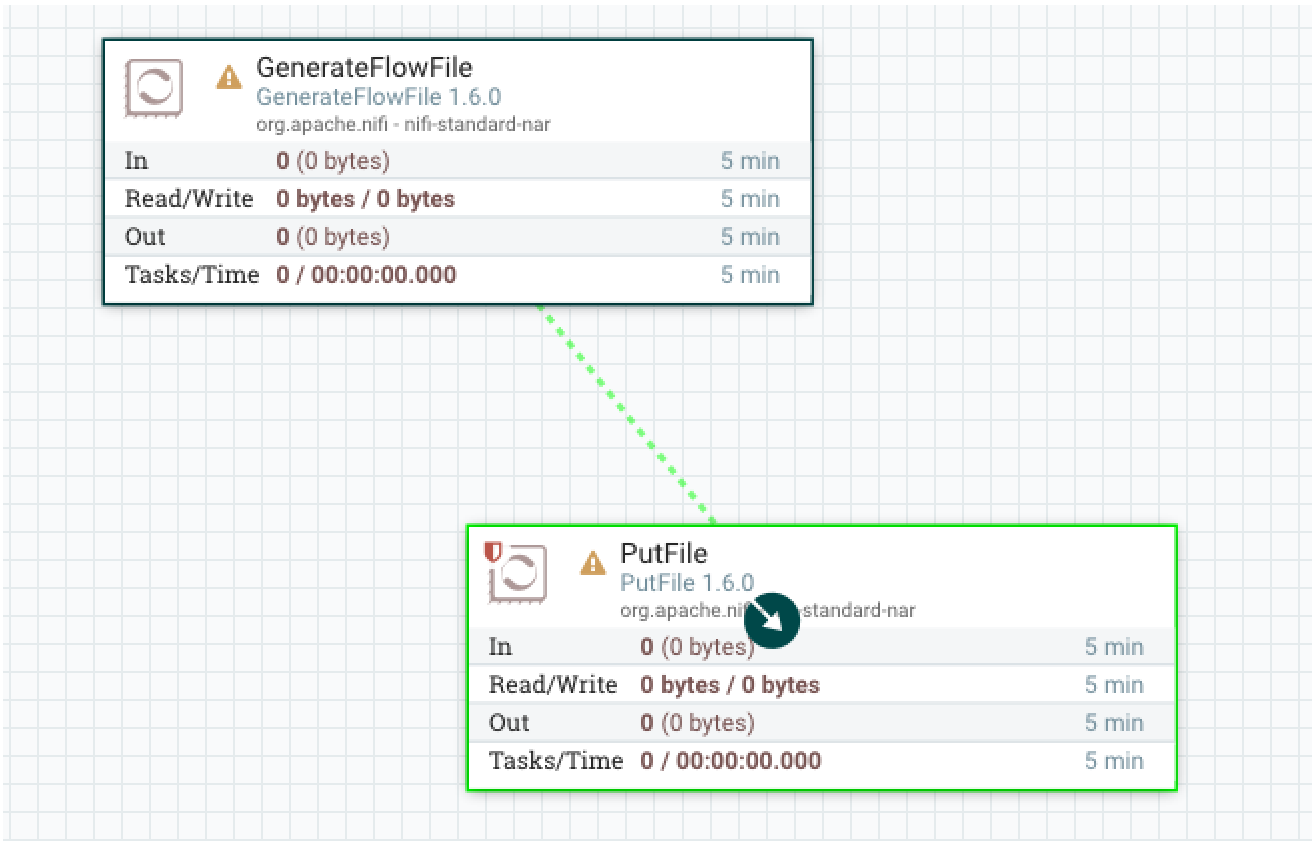
Yield Duration
1 sec

Bulletin Level
WARN

Automatically Terminate Relationships
☑ failure
Files that could not be written to the output directory for some reason are transferred to this relationship
☑ success
Files that have been successfully written to the output directory are transferred to this relationship

Step 5 - Connect the processors

From `GenerateFile` to `Putfile`:

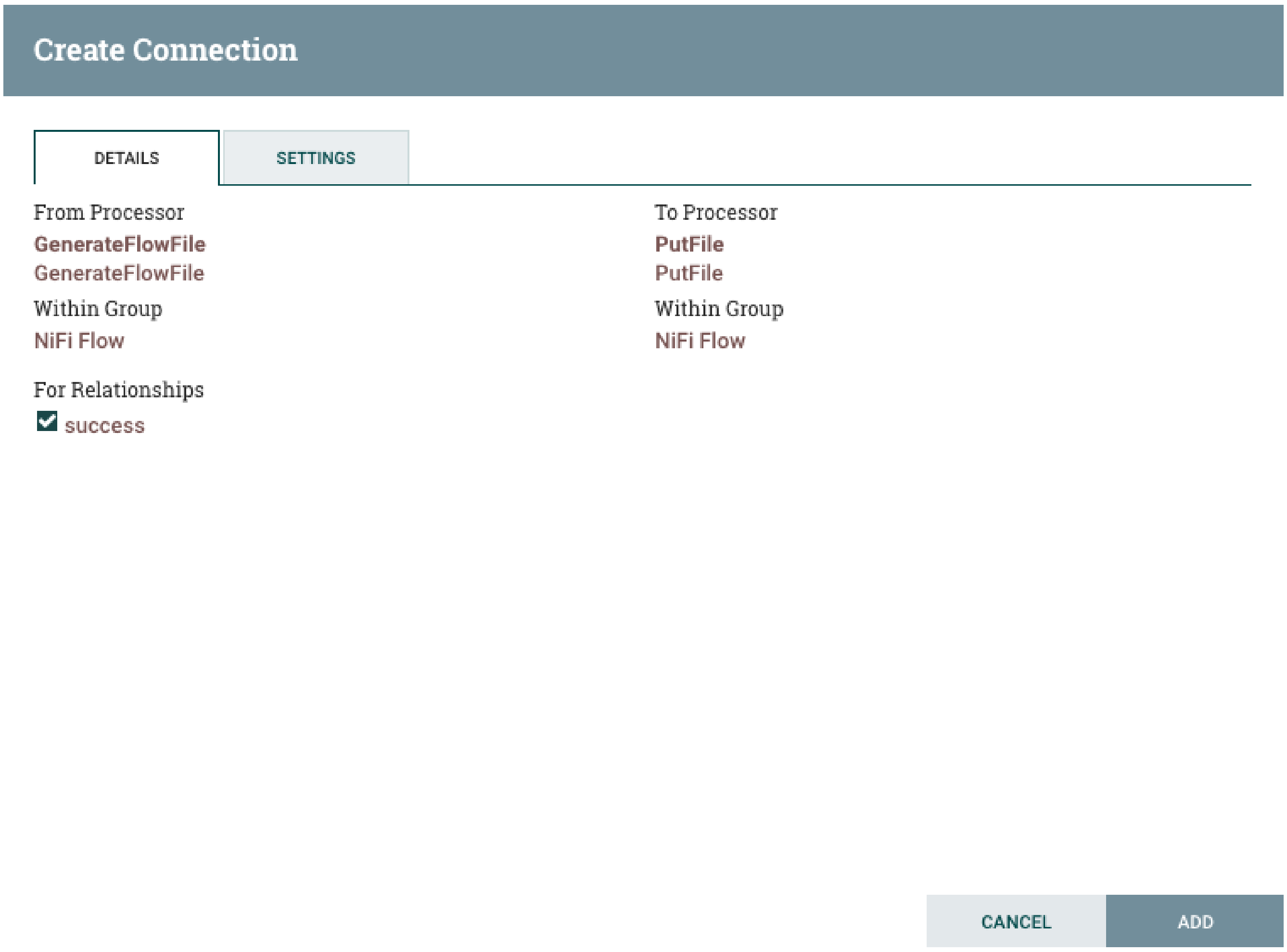

GenerateFlowFile
GenerateFlowFile 1.6.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

PutFile
PutFile 1.6.0
org.apache.ni standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

A connection will be create:


Create Connection

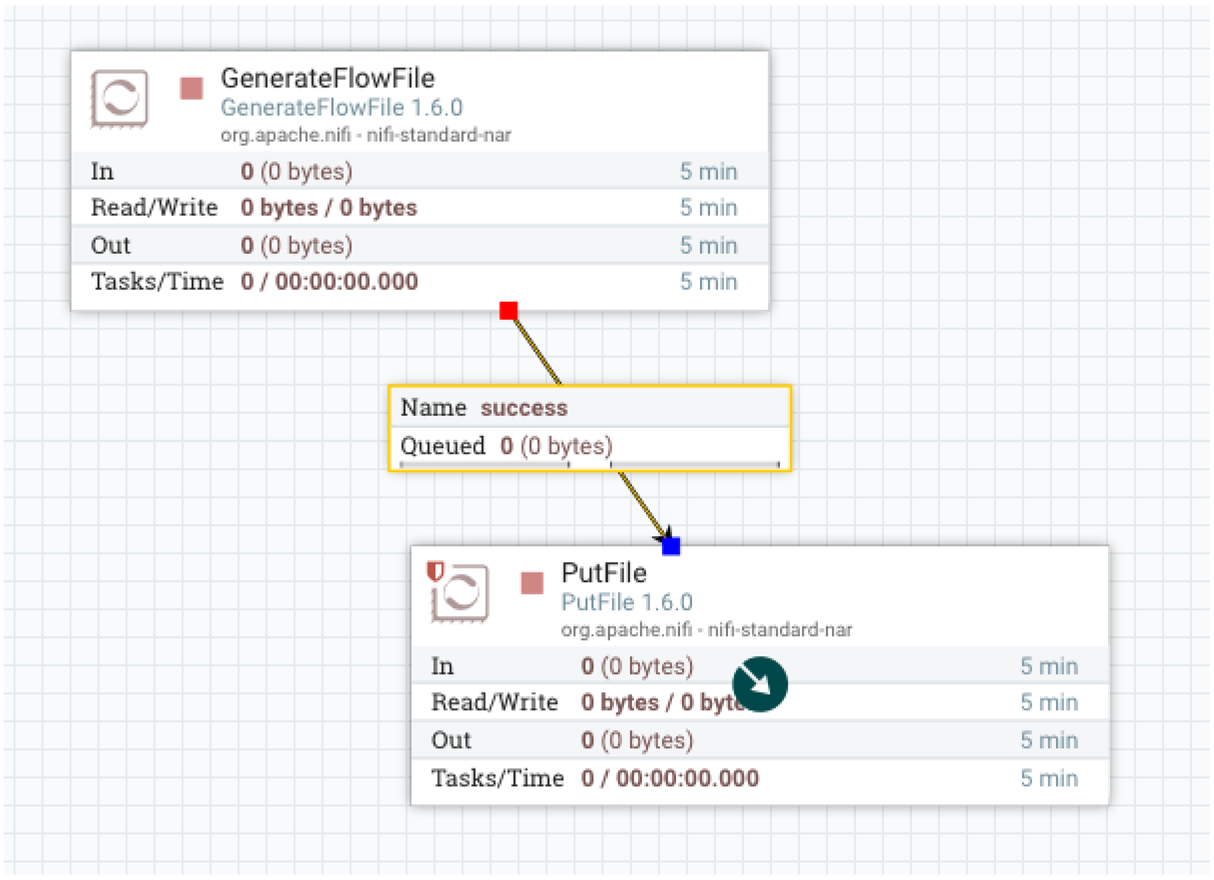
DETAILS | SETTINGS

From Processor
GenerateFlowFile
GenerateFlowFile
Within Group
NiFi Flow

To Processor
PutFile
PutFile
Within Group
NiFi Flow

For Relationships
✓ success

This will be the final state:


GenerateFlowFile
GenerateFlowFile 1.6.0
org.apache.nifi - nifi-standard-nar

```
In           0 (0 bytes)              5 min
Read/Write   0 bytes / 0 bytes        5 min
Out          0 (0 bytes)              5 min
Tasks/Time   0 / 00:00:00.000         5 min
```

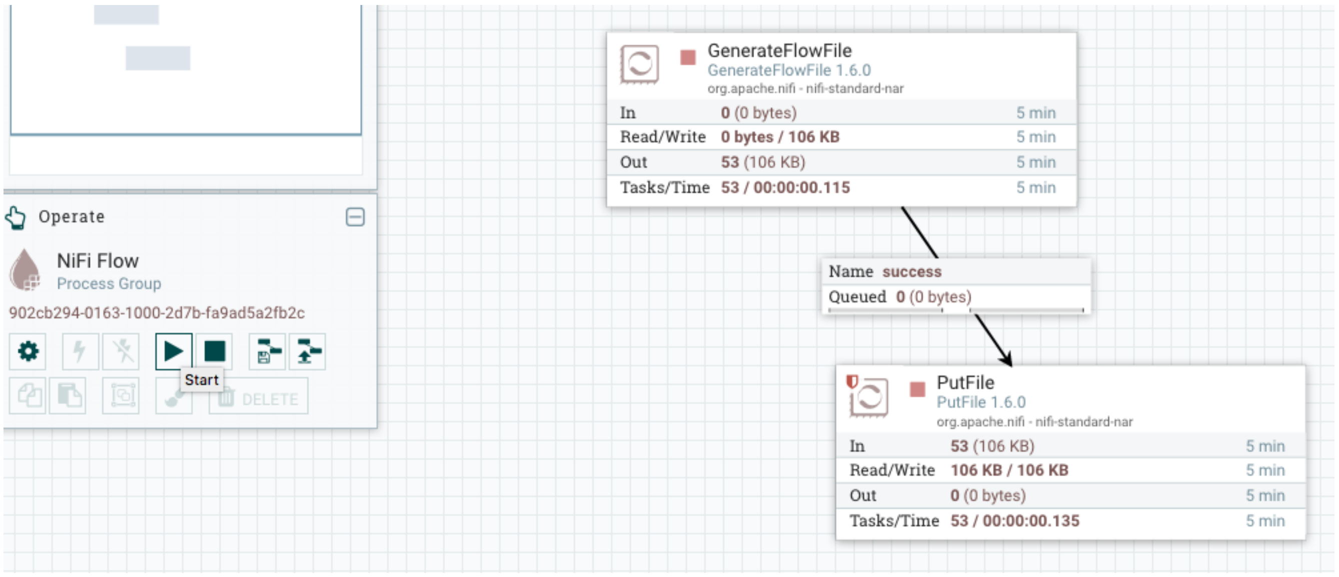
```
Name    success
Queued  0 (0 bytes)
```

PutFile
PutFile 1.6.0
org.apache.nifi - nifi-standard-nar

```
In           0 (0 bytes)              5 min
Read/Write   0 bytes / 0 byt          5 min
Out          0 (0 bytes)              5 min
Tasks/Time   0 / 00:00:00.000         5 min
```

Step 6 - Start the processors

Click `Ctrl` to select both processors and start it.



GenerateFlowFile
GenerateFlowFile 1.6.0
org.apache.nifi - nifi-standard-nar

```
In           0 (0 bytes)              5 min
Read/Write   0 bytes / 106 KB         5 min
Out          53 (106 KB)              5 min
Tasks/Time   53 / 00:00:00.115        5 min
```

```
Name    success
Queued  0 (0 bytes)
```

PutFile
PutFile 1.6.0
org.apache.nifi - nifi-standard-nar

```
In           53 (106 KB)              5 min
Read/Write   106 KB / 106 KB          5 min
Out          0 (0 bytes)              5 min
Tasks/Time   53 / 00:00:00.135        5 min
```

Step 7 - View the generated logs

Open a shell inside the container:

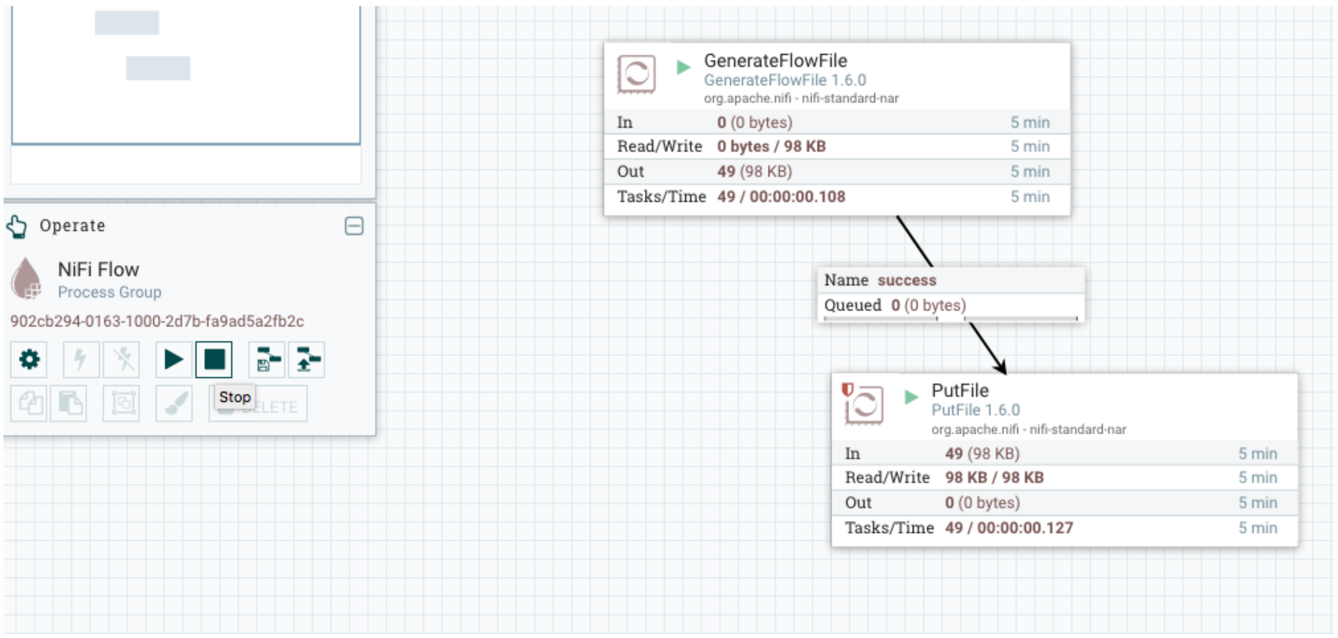
```
$ docker exec -it nifi /bin/bash
```

Type the following command to see a list of the 9 generated files. This list will be actualized second by second. As we configure in NiFi, a new file will be generated on every 5 seconds.

```
$ while :; do clear; ls -lht /tmp/nifi/ | head -10; sleep 1; done
```

Step 8 - Stop the processors

Click `Ctrl` to select both processors and stop it.



GenerateFlowFile
GenerateFlowFile 1.6.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 98 KB	5 min
Out	49 (98 KB)	5 min
Tasks/Time	49 / 00:00:00.108	5 min

Name success
Queued 0 (0 bytes)

PutFile
PutFile 1.6.0
org.apache.nifi - nifi-standard-nar

In	49 (98 KB)	5 min
Read/Write	98 KB / 98 KB	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	49 / 00:00:00.127	5 min

Step 9 - Stop and destroy the docker container

```
$ docker stop nifi
$ docker rm nifi
```

Conclusions

- NiFi UI is very simple and intuitive.
- The properties are well documented.
- Many other aspects of the UI can be explored in this playlist.

All references

Apache NiFi

GitHub

- apache/nifi
- nifi-docker/dockerhub

Documentation

- Apache NiFi User Guide
- NiFi Developer's Guide
- Apache NiFi In Depth

Other

YouTube videos

- yA: Matt Burgess discusses Open Source Software & Apache nifi
- yB: Spring with ApacheNiFi
- yC: Hortonworks DataFlow powered by Apache NiFi
- yD: How to navigate and build a dataflow in Apache Nifi
- yE: Apache NiFi How to Build a Flow - Part 1 (Updated)
- yF: Apache NiFi How to Build a Flow - Part 2 (Updated)
- Videos page at Silver Cloud Computing

GitHub

- hortonworks-gallery/nifi-templates
- ndstreev/nifi-processor-examples
- aperepel/nifi-api-deploy

Community Forums/Meetups

- http://apache-nifi.1125220.n5.nabble.com/
- https://community.hortonworks.com/topics/Nifi.html

Stack Overflow

- automating NIFI template deployment

Books

- NiFi Fundamentals & Cookbook: 9-Use cases, covering various scenarios

Articles/ examples

- Getting Started
  - Getting Started with Apache Nifi
  - Apache NiFi: An Introduction

    - NiFi page at Silver Cloud Computing
    - Hello-World in Apache NiFi
    - Hello NiFi! – Data Orchestration using Hortonworks DataFlow (HDF)
- Developing custom processors
    - Developing A Custom Apache Nifi Processor (JSON)
    - Building a Custom Processor in Apache NiFi 1.5 for TensorFlow Using the Java API
- Automation/ Deployment
    - Apache NiFi - How do I deploy my flow?
    - Automate workflow deployment in Apache NiFi with the NiFi Registry
- Best Practices/ Use cases
    - Best practices for using Apache NiFi in real world projects - 3 takeaways
    - Best practices and lessons learnt from Running Apache NiFi at Renault
- Architecture
    - Apache Nifi Architecture
- Events/ Meetups
    - https://www.meetup.com/en-AU/Hadoop-User-Group-Vienna/events/248266228/
    - Cloud Operations with Streaming Analytics using Apache Nifi and Apache Flink
- Spring with Apache NiFi:
    - Spring with ApacheNiFi
- Pivotal Cloud Foundry integration:
    - Apache NIFI as a PCF service
- Comparison with other tools:
    - What are the main differences between Spring Cloud Data Flow and Apache Nifi?