Spring Cloud Data Flow Lab

Setting up the environment

1. Install RabbitMQ following the instructions in the document <FOLDER>/DNDataflow/labs/InstallRabbitMQ.pdf.
2. Now let's setup the lab environment:
    a. Using git
        i. From a your local terminal or command prompt change directory to a clean working directory.
        ii. Now execute: `git clone https://github.com/cppwfs/DNDataflow.git`
        iii. Now `cd DNDataflow`
    b. Using Thumbdrive
        i. Copy the **DNDataflow** directory from the thumbdrive to a location on your laptop hard drive
        ii. Now from a terminal or command prompt **cd** to the **DNDataflow** directory you just created on your hard drive.

Creating your first stream

- If you haven't already completed all the installation steps from `lab3`, please proceed there to set up Spring Cloud Data Flow "Server" and "Shell" applications
- Register applications from Shell application

(1) Source

```
app register --name http --type source --uri
file:///<FOLDER>/DNDataflow/labs/jars/http-source-rabbit-1.1.2.RELEASE.jar
```

(2) Processor

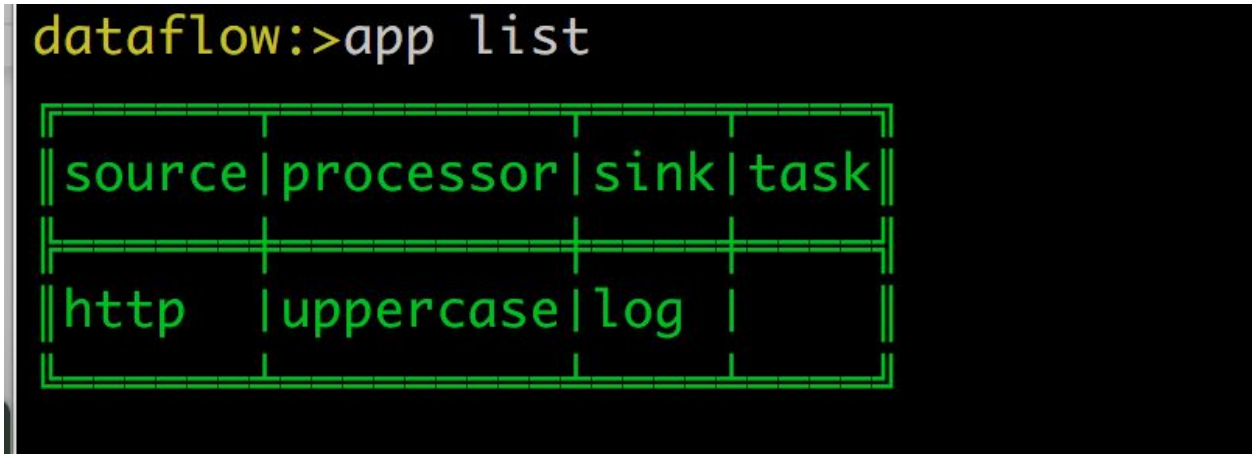
```
app register --name uppercase --type processor --uri
file:///<FOLDER>/DNDataflow/labs/jars/streamlab-0.0.1-SNAPSHOT.jar
```

(3) Sink

```
app register --name log --type sink --uri
file:///<FOLDER>/DNDataflow/labs/jars/log-sink-rabbit-1.1.1.RELEASE.jar
```

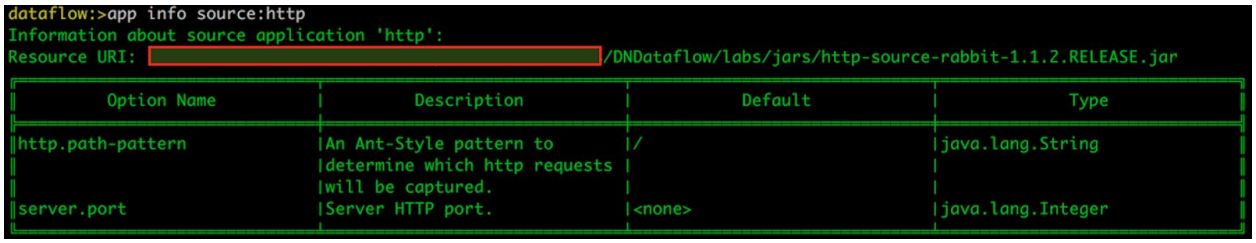
- List the registered applications

```
app list
```



- Verify the registered applications

```
app info source:http
```



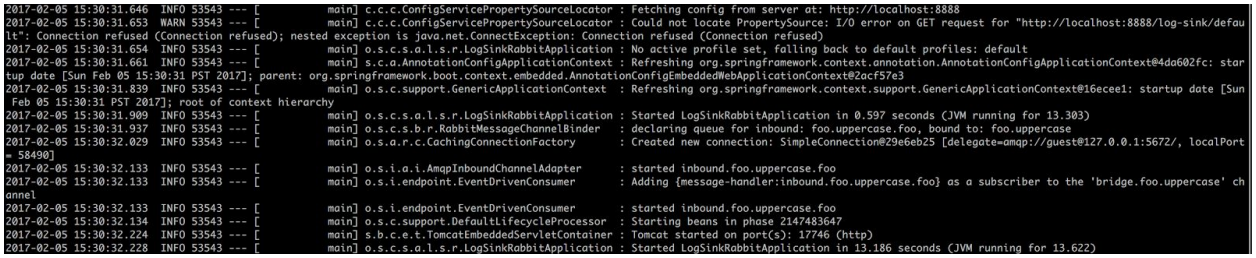
- Create a stream

```
stream create foo --definition "http --port=9001 | uppercase | log"
--deploy
```

- Tail the log-sink logs; for example:

In the server console, you will see both the `http-source` and `log-sink` logs being logged to at a special directory. Copy the path for “foo.log” application and use the “tail” command to review the logs like the following. If you are a Windows user, you could either use something like Cygwin or open the file in a text-editor and that could load and refresh the contents continuously.

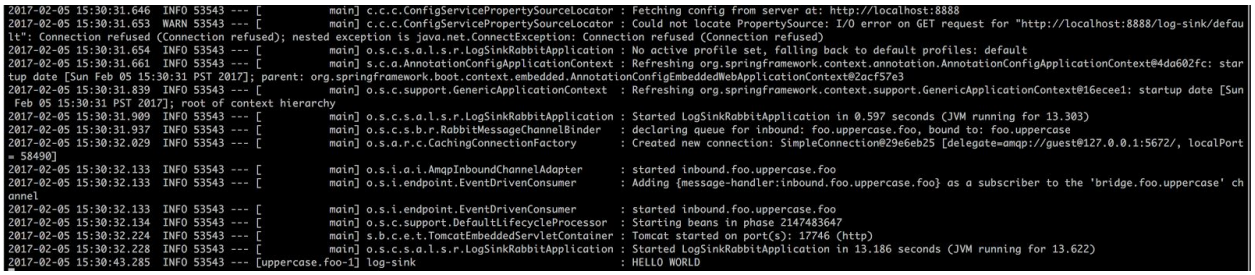
```
tail -f
/var/folders/c3/ctx7_rns6x30tq7rb76wzqwr0000gp/T/spring-cloud-dataflo
w-3545000607490975505/foo-1486337156762/foo.log/stdout_0.log
```



- Post some data against the target http://localhost:9001

```
dataflow:>http post --target http://localhost:9001 --data "hello world"
> POST (text/plain;Charset=UTF-8) http://localhost:9001 hello world
> 202 ACCEPTED
```

- Verify the log-sink logs for "HELLO WORLD"



- Destroy the stream

```
dataflow:>stream destroy foo
```

Partitioned Stream

- We will use a new type of processor in this exercise. A splitter-processor, as you might have assumed, it splits the payload by the specified character. Let's register this application.

```
app register --name splitter --type processor --uri
file:///<FOLDER>/DNDataflow/labs/jars/splitter-processor-rabbit-1.1.1.RELEASE.jar
```

If you haven't already, please also register http-source and log-sink applications.

```
app register --name http --type source --uri
file:///<FOLDER>/DNDataflow/labs/jars/http-source-rabbit-1.1.2.RELEASE.jar
```

```
app register --name log --type sink --uri
file:////<FOLDER>/DNDataflow/labs/jars/log-sink-rabbit-1.1.1.RELEASE.jar
```

- Create a simple partitioned stream

```
stream create --name words --definition "http --server.port=9900 |
splitter --expression=payload.split(' ') | log"
```

- Deploy the partitioned stream with 2 instances of log-sink

```
stream deploy words --properties
"app.splitter.producer.partitionKeyExpression=payload,app.log.count=2
"
```

- Tail both the log-sink instances; for example:

In the server console, you will see both the `http-source` and `log-sink` logs being logged to a special directory. Copy the path for "`words.log`" application and use the "tail" command to review the logs like the following. There will be 2 instances of this log file; one from each of the `log-sink` application instance. If you are a Windows user, you could either use something like Cygwin or open the file in a text-editor and that could load and refresh the contents continuously.

```
tail -f
/var/folders/c3/ctx7_rns6x30tq7rb76wzqwr0000gp/T/spring-cloud-dataflo
w-3545000607490975505/words-1486337773441/words.log/stdout_0.log

tail -f
/var/folders/c3/ctx7_rns6x30tq7rb76wzqwr0000gp/T/spring-cloud-dataflo
w-3545000607490975505/words-1486337773441/words.log/stdout_1.log
```

- Post the following data

```
dataflow:>http post --target http://localhost:9900 --data "How much
wood would a woodchuck chuck if a woodchuck could chuck wood"
> POST (text/plain;Charset=UTF-8) http://localhost:9900 How much wood
would a woodchuck chuck if a woodchuck could chuck wood
> 202 ACCEPTED
```

- Observe the log-sink logs

words.log instance 0

```
2017-02-05 15:36:21.999  INFO 53948 --- [           main] o.s.c.s.b.r.RabbitMessageChannelBinder   : declaring queue for inbound: words.splitter.words, bound to: words.splitter
2017-02-05 15:36:22.091  INFO 53948 --- [           main] o.s.a.r.c.CachingConnectionFactory       : Created new connection: SimpleConnection@1ff8c153 [delegate=amqp://guest@127.0.0.1:5672/]
2017-02-05 15:36:22.175  INFO 53948 --- [           main] o.s.integration.channel.DirectChannel    : Channel 'words.splitter.words.bridge' has 1 subscriber(s).
2017-02-05 15:36:22.189  INFO 53948 --- [           main] o.s.i.a.i.AmqpInboundChannelAdapter      : started inbound.words.splitter.words
2017-02-05 15:36:22.189  INFO 53948 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 2147483647
2017-02-05 15:36:22.191  INFO 53948 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 0
2017-02-05 15:36:22.191  INFO 53948 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 2147482647
2017-02-05 15:36:22.191  INFO 53948 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 2147483647
2017-02-05 15:36:22.302  INFO 53948 --- [           main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 24451 (http)
2017-02-05 15:36:22.307  INFO 53948 --- [           main] o.s.c.s.a.l.s.r.LogSinkRabbitApplication : Started LogSinkRabbitApplication in 8.306 seconds (JVM running for 8.85)
2017-02-05 15:36:59.125  INFO 53948 --- [itter.words-0-1] log.sink                                 : How
2017-02-05 15:36:59.130  INFO 53948 --- [itter.words-0-1] log.sink                                 : chuck
2017-02-05 15:36:59.132  INFO 53948 --- [itter.words-0-1] log.sink                                 : chuck
```

words.log instance 1

```
2017-02-05 15:36:21.816  INFO 53949 --- [           main] o.s.c.s.b.r.RabbitMessageChannelBinder   : declaring queue for inbound: words.splitter.words, bound to: words.splitter
2017-02-05 15:36:21.911  INFO 53949 --- [           main] o.s.a.r.c.CachingConnectionFactory       : Created new connection: SimpleConnection@4005970f [delegate=amqp://guest@127.0.0.1:5672/]
2017-02-05 15:36:21.996  INFO 53949 --- [           main] o.s.integration.channel.DirectChannel    : Channel 'words.splitter.words.bridge' has 1 subscriber(s).
2017-02-05 15:36:22.010  INFO 53949 --- [           main] o.s.i.a.i.AmqpInboundChannelAdapter      : started inbound.words.splitter.words
2017-02-05 15:36:22.010  INFO 53949 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 2147483647
2017-02-05 15:36:22.011  INFO 53949 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 0
2017-02-05 15:36:22.011  INFO 53949 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 2147482647
2017-02-05 15:36:22.012  INFO 53949 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 2147483647
2017-02-05 15:36:22.126  INFO 53949 --- [           main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 15918 (http)
2017-02-05 15:36:22.129  INFO 53949 --- [           main] o.s.c.s.a.l.s.r.LogSinkRabbitApplication : Started LogSinkRabbitApplication in 7.76 seconds (JVM running for 8.433)
2017-02-05 15:36:59.120  INFO 53949 --- [itter.words-1-1] log.sink                                 : much
2017-02-05 15:36:59.123  INFO 53949 --- [itter.words-1-1] log.sink                                 : wood
2017-02-05 15:36:59.125  INFO 53949 --- [itter.words-1-1] log.sink                                 : would
2017-02-05 15:36:59.127  INFO 53949 --- [itter.words-1-1] log.sink                                 : a
2017-02-05 15:36:59.129  INFO 53949 --- [itter.words-1-1] log.sink                                 : woodchuck
2017-02-05 15:36:59.131  INFO 53949 --- [itter.words-1-1] log.sink                                 : if
2017-02-05 15:36:59.133  INFO 53949 --- [itter.words-1-1] log.sink                                 : a
2017-02-05 15:36:59.136  INFO 53949 --- [itter.words-1-1] log.sink                                 : woodchuck
2017-02-05 15:36:59.138  INFO 53949 --- [itter.words-1-1] log.sink                                 : could
2017-02-05 15:36:59.140  INFO 53949 --- [itter.words-1-1] log.sink                                 : wood
```