Spring Cloud Stream Lab

# Setting up the environment

1. Install RabbitMQ following the instructions in **Lab 0 - Install RabbitMQ.**
2. Now let’s setup the lab environment:
   1. Using git
      1. From a your local terminal or command prompt change directory to a clean working directory.
      2. Now execute: git clone https://github.com/cppwfs/DNDataflow.git
      3. Now cd DNDataflow
   2. Using Thumbdrive
      1. Copy the **DNDataflow** directory from the thumbdrive to a location on your laptop hard drive
      2. Now from a terminal or command prompt **cd** to the **DNDataflow** directory you just created on your hard drive.

# Creating your Processor

* CD into “DNDataflow/labs/lab1” folder
* First let’s review the project in your favourite IDE/text-editor
* Start with the pom.xml and review the project dependencies
* You’ll find the following dependency that brings the binder implementation we would like to use in this lab, which happens to be a RabbitMQ implementation in our case. (*You’ll also find few other dependencies that are commented out; let’s ignore them for now*)

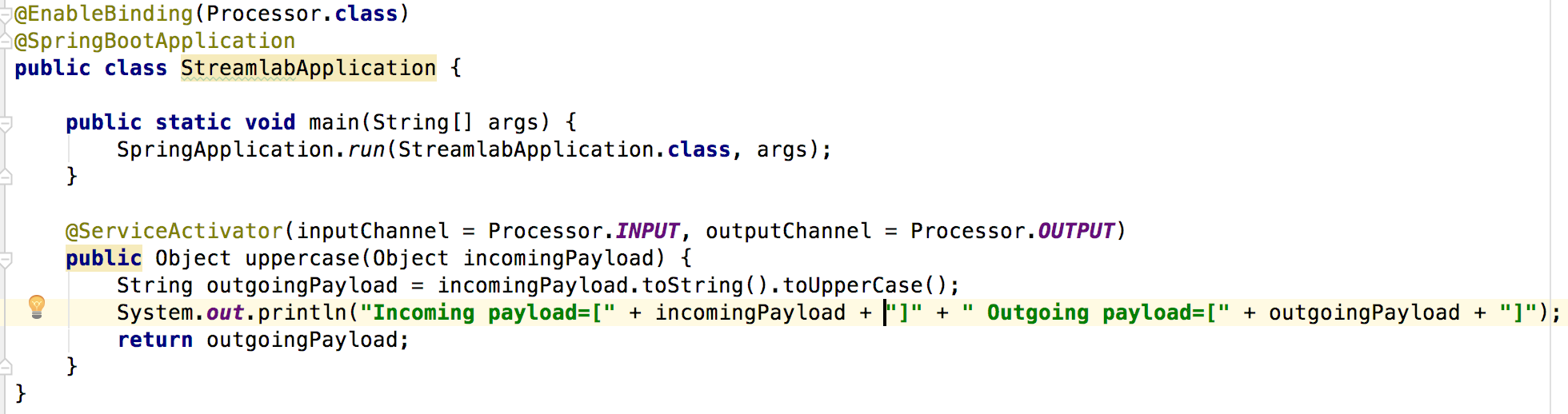
<**dependency**>

<**groupId**>org.springframework.cloud</**groupId**>

<**artifactId**>spring-cloud-starter-stream-rabbit</**artifactId**>

</**dependency**>

* Open “StreamlabApplication” and add @EnableBinding(Processor.class) annotation
* Uncomment the uppercase() business logic and review the INPUT and OUTPUT channel configurations activated via @ServiceActivator



* From the terminal/shell/command-prompt, let’s build the application

mvnw clean package

* Run the application and verify it starts normally

java -jar target/streamlab-0.0.1-SNAPSHOT.jar

* Uncomment Spring Boot’s Actuator dependency

<**dependency**>

<**groupId**>org.springframework.boot</**groupId**>

<**artifactId**>spring-boot-starter-actuator</**artifactId**>

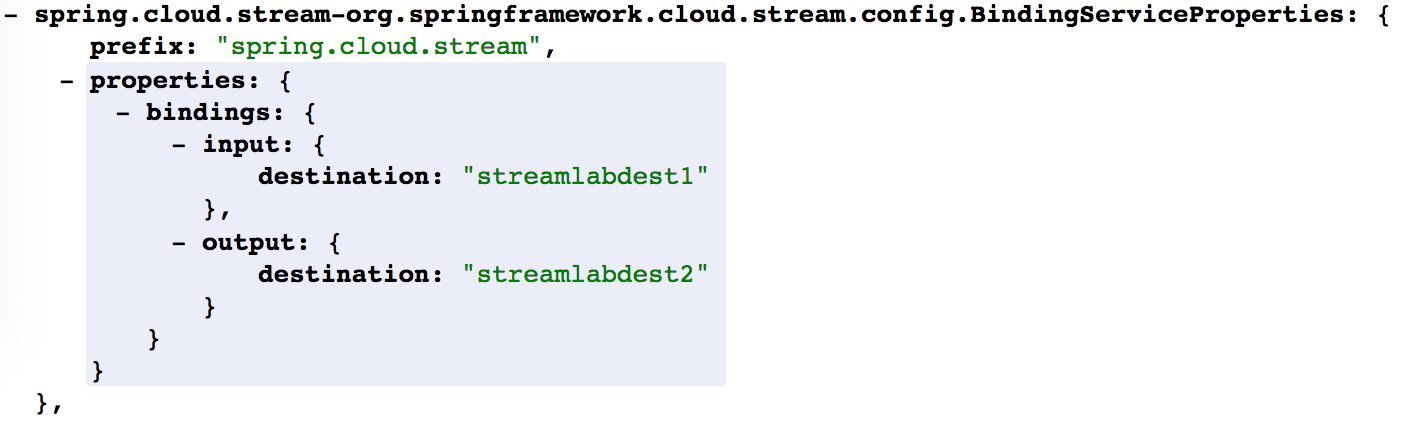
</**dependency**>

* Rebuild and run the application to review the actuator endpoints; specifically, we would want to confirm whether the channel bindings are setup correctly

java -jar target/streamlab-0.0.1-SNAPSHOT.jar

* Launch “/configprops” endpoint and verify the channel bindings loaded from the application.properties file

http://localhost:9002/configprops



* Now, let’s build a streaming pipeline

**(1) Source**

java -jar labs/jars/http-source-rabbit-1.1.2.RELEASE.jar --spring.cloud.stream.bindings.output.destination=**streamlabdest1** --server.port=9001

**(2) Processor**

java -jar labs/lab1/target/streamlab-0.0.1-SNAPSHOT.jar --spring.cloud.stream.bindings.input.destination=**streamlabdest1** --spring.cloud.stream.bindings.output.destination=**streamlabdest2** --server.port=9002

**(3) Sink**

java -jar labs/jars/log-sink-rabbit-1.1.1.RELEASE.jar --spring.cloud.stream.bindings.input.destination=**streamlabdest2** --server.port=9003

* Monitor the logs of all the 3 applications in different terminal windows
* Post data to the http://localhost endpoint at port=9001

curl -target http://localhost:9001 -H "Content-Type:text/plain" -d "hello world"

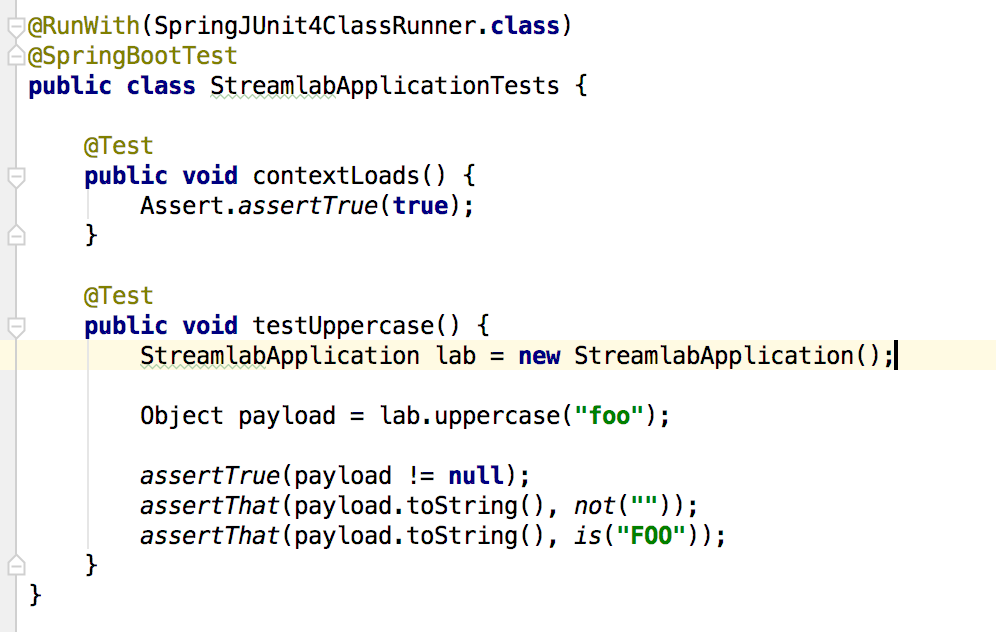
* Review the the logs in the log-sink console

# 

# Testing your Processor Code

Both unit and integration tests are included in the `/lab1` project

* Unit Test
  + Uncomment “StreamlabApplicationTests”



* + Change the expectations and let the test fail
  + Run the tests
  + Verify the assertions fail
  + Revert to original state
  + Run the tests
  + Verify the assertions work correctly
* Integration Test
  + Uncomment “spring-cloud-stream-test-support*”* from pom.xml

*<!--<dependency>-->*

*<!--<groupId>org.springframework.cloud</groupId>-->*

*<!--<artifactId>spring-cloud-stream-test-support</artifactId>-->*

*<!--</dependency>-->*

* + Re-import/refresh maven dependencies at the project level
  + Uncomment “StreamlabIntegrationTests”



* + Change the expectations and let the test fail
  + Run the tests
  + Verify the assertions fail
  + Revert to original state
  + Run the tests
  + Verify the assertions work correctly

For the curious, who would like to start from Spring Initializr experience, please select the dependencies as listed below and then generate a new project.

