Spring Cl/ud Stream Lab

Setting up the envir/nment

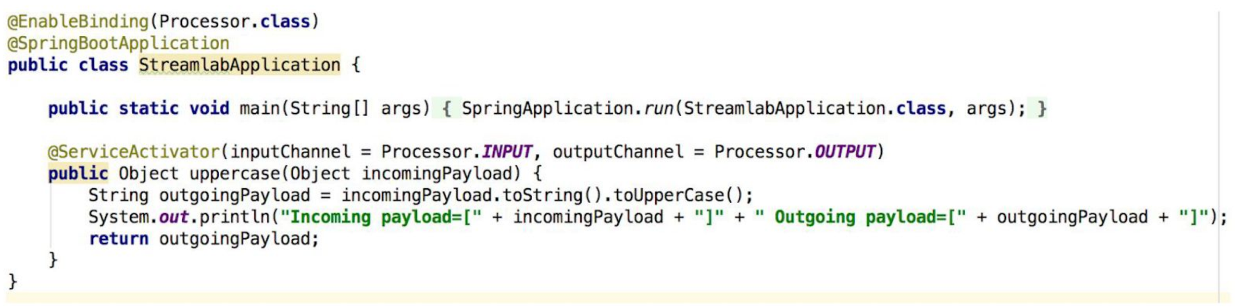
1. Install RabbitMQ f/ll/wing the instructi/ns in Lab 0 - Install RabbitMQÆ
2. N/w let's setup the lab envir/nment:
   a. Using git
      i. Fr/m a y/ur l/cal terminal /r c/mmand pr/mpt change direct/ry t/ a clean w/rking direct/ry.
      ii. N/w execute: git cl/ne https://github.c/m/wxlund/DNDatafl/w
      iii. N/w cd DNDatafl/w
   b. Using Thumb drive (N/te: we haven't created a zip /f the cl/ne yet. Determine if lab will need this.
      i. C/py the DNDatafl/w direct/ry fr/m the thumb drive t/ a l/cati/n /n y/ur lapt/p hard drive
      ii. N/w fr/m a terminal /r c/mmand pr/mpt cd t/ the DNDatafl/w direct/ry y/u just created /n y/ur hard drive.

Creating y/ur Pr/cess/r

- CD int/ "DNDatafl/w/labs/lab1" f/lder
- First let's review the pr/ject in y/ur fav/urite IDE/text-edit/r
- Start by /pening the p/m.xm and review the pr/ject dependencies
- Y/u'll find the f/ll/wing dependency that brings the binder implementati/n we w/uld like t/ use in this lab, which happens t/ be a RabbitMQ implementati/n in /ur case. *(Y/u'll als/ find few /ther dependencies that are c/mmented /utª let's ign/re them f/r n/w)*

```
<dependency>
        <gr/upId> /rg.springframew/rk.cl/ud </gr/upId>
        <artifactId>spring-cl/ud-starter-stream-rabbit</artifactId>
</dependency>
```

- /pen " StreamlabApplicati/n " and add @EnableBinding(Pr/cess/r/class) ann/tati/n

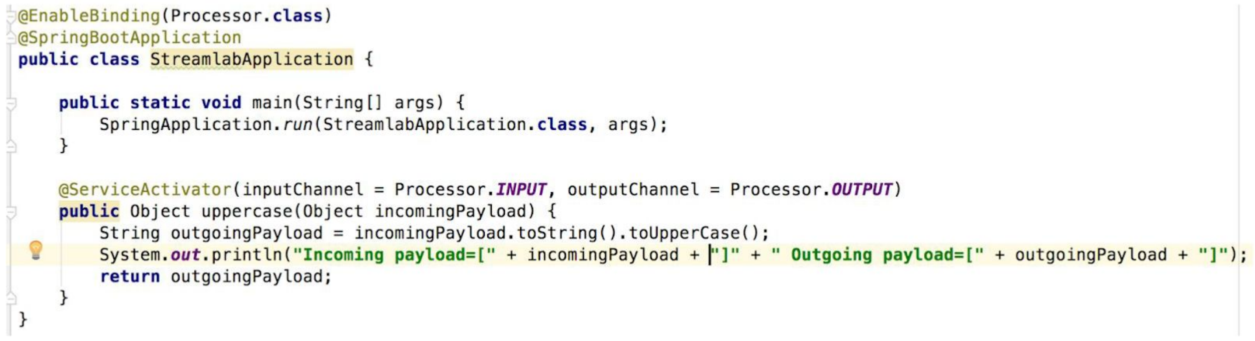


```java
@EnableBinding(Processor.class)
@SpringBootApplication
public class StreamlabApplication {

    public static void main(String[] args) { SpringApplication.run(StreamlabApplication.class, args); }

    @ServiceActivator(inputChannel = Processor.INPUT, outputChannel = Processor.OUTPUT)
    public Object uppercase(Object incomingPayload) {
        String outgoingPayload = incomingPayload.toString().toUpperCase();
        System.out.println("Incoming payload=[" + incomingPayload + "]" + " Outgoing payload=[" + outgoingPayload + "]");
        return outgoingPayload;
    }
}
```

- Unc/mment the uppercase () business l/gic and review the INPUT and /UTPUT channel c/nfigurati/ns activated via @ServiceActivat/r



```
@EnableBinding(Processor.class)
@SpringBootApplication
public class StreamlabApplication {

    public static void main(String[] args) {
        SpringApplication.run(StreamlabApplication.class, args);
    }

    @ServiceActivator(inputChannel = Processor.INPUT, outputChannel = Processor.OUTPUT)
    public Object uppercase(Object incomingPayload) {
        String outgoingPayload = incomingPayload.toString().toUpperCase();
        System.out.println("Incoming payload=[" + incomingPayload + "]" + " Outgoing payload=[" + outgoingPayload + "]");
        return outgoingPayload;
    }
}
```

- Fr/m the terminal/shell/c/mmand-pr/mpt, let's build the applicati/n

```
mvnw clean package
```

- Run the applicati/n and verify it starts n/rmally

```
java  -jar target/streamlab-0.0.1-SNAPSH/T.jar
```

- Unc/mment Spring B//t's Actuat/r dependency

```
<dependency>
        <gr/upId> /rg.springframew/rk.b//t</gr/upId>
        <artifactId> spring-b//t-starter-actuat/r</artifactId>
</dependency>
```

- Rebuild and run the applicati/n t/ review the actuat/r endp/ints; specifically, we w/uld want t/ c/nfirm whether the channel bindings are setup c/rrectly

```
java  -jar target/streamlab-0-0-1-SNAPSH/T.jar
```

- Launch "http://<H/ST>:<P/RT>/ c/nfigpr/ps " endp/int and verify the channel bindings l/aded fr/m the applicati/nÆpr/perties

  Example:

```
file http://l/calh/st:9002/c/nfigpr/ps
```

- N/w, let's build a streaming pipeline

(1) Source

This app is a simple spring boot application and is available from the app starters. For convenience a copy of the jar has been moved into the labs/jars directory. The app-starters are found at https://repo.spring.io/libs-release/org/springframework/cloud/stream/app/spring-cloud-stream-app-descriptor/Einstein.RELEASE/spring-cloud-stream-app-descriptor-Einstein.RELEASE.rabbit-apps-maven-repo-url.properties in our configuration of using rabbit.

```
java -jar labs/jars/http-source-rabbit-2.1.0.RELEASE.jar
-spring.cloud.stream.bindings.output.destination=streamlabdest1
--server.port=9001
```

(2) Processor

This is the app that you built, which is why the jar is found from the labs /target directory

```
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

This app is a simple spring boot application and is available from the app starters. See the note from (1) Source.

```
java -jar labs/jars/log-sink-rabbit-2.1.0.RELEASE.jar
--spring.cloud.stream.bindings.input.destination=streamlabdest2
--server.port=9003
```

- Monitor the logs of all the 3 applications in the terminal window where the app was launched.
- Post data to the http://localhost endpoint at port=9001

```
curl  -target https://localhost:9001 -H “Content-Type:text/plain”  -d
“hello world”
```

● Review the the logs in the log-sink console (terminal)

Testing your Processor

B/th unit and integrati/n tests are included in the `/lab1` pr/ject

● Unit Test

○ Unc/mment “ StreamlabApplicati/nTests ”

○ Change the expectati/ns and let the test fail

○ Run the tests

○ Verify the asserti/ns fail

○ Revert t/ /riginal state

○ Run the tests

○ Verify the asserti/ns w/rk c/rrectly

● Integrati/n Test

○ Unc/mment “ spring-cl /ud-stream-test-supp/rt” fr/m p/mÆxml

º° - -ºdependencyæ - -æ

º° - -ºgr/upIdæ/rgÆspringframew/rkÆcl/udº/gr/upIdæ - -æ

º° - -ºartifactIdæspring -cl/ud -stream -test -supp/rtº/artifactIdæ - -æ

º° - -º/dependencyæ - -æ

○ Re-imp/rt/refresh maven dependencies at the pr/ject level

○ Unc/mment “ StreamlabIntegrati/nTests ”

○ Change the expectati/ns and let the test fail

○ Run the tests

○ Verify the asserti/ns fail

○ Revert t/ /riginal state

○ Run the tests

○ Verify the asserti/ns w/rk c/rrectly

Appendix

F/r the curi/us, wh/ w/uld like t/ start fr/m Spring Initializr experience, please select the dependencies as listed bel/w and then generate a new pr/ject.