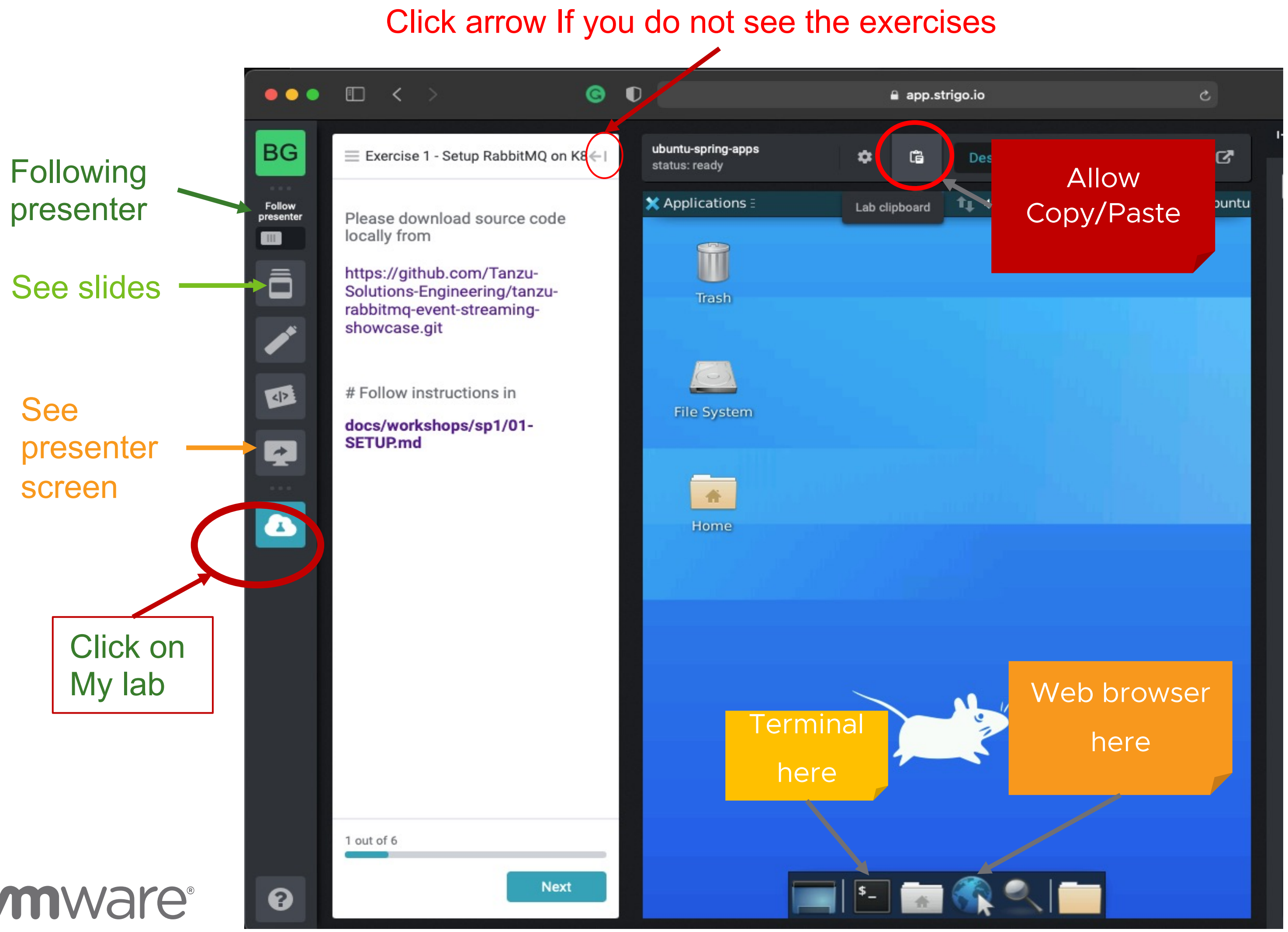
Never Lose a Message Again: Building Rock-Solid Event-Driven Streaming Architectures with

VMware Tanzu RabbitMQ and Spring

Getting Started

Download Source Code

https://github.com/Tanzu-Solutions-Engineering/tanzu-rabbitmq-event-streaming-showcase.git



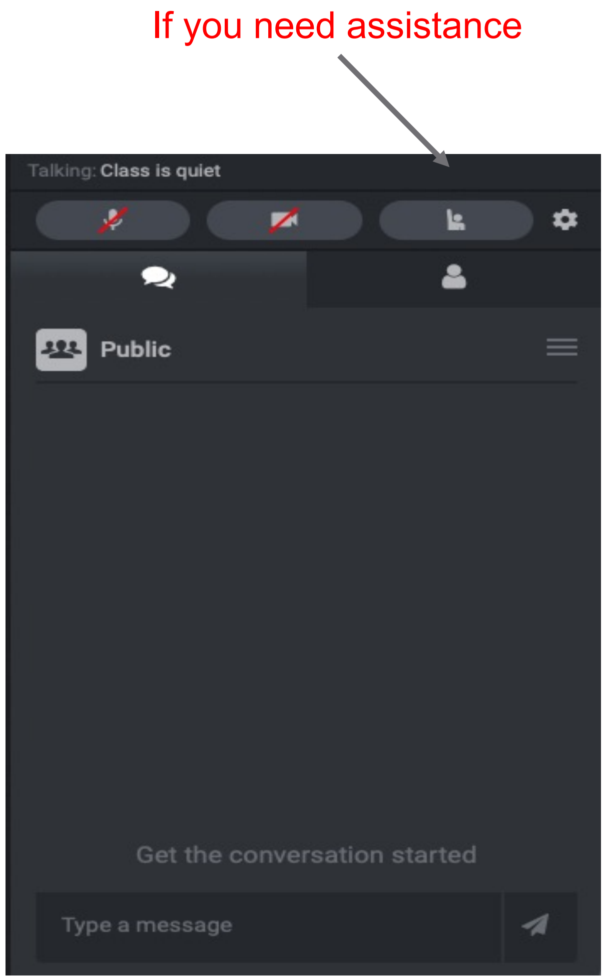
Exercise 1 - Setup RabbitMQ on K8

Please download source code locally from

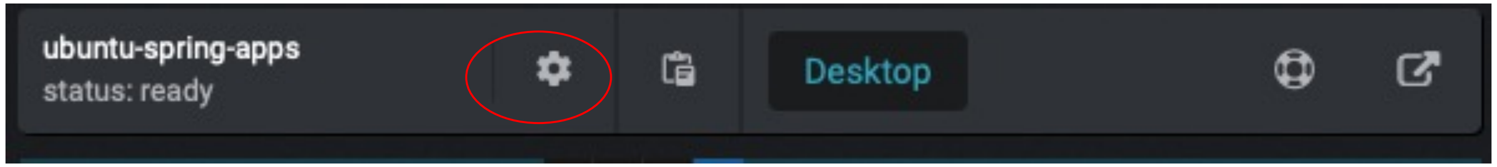
https://github.com/Tanzu-Solutions-Engineering/tanzu-rabbitmq-event-streaming-showcase.git

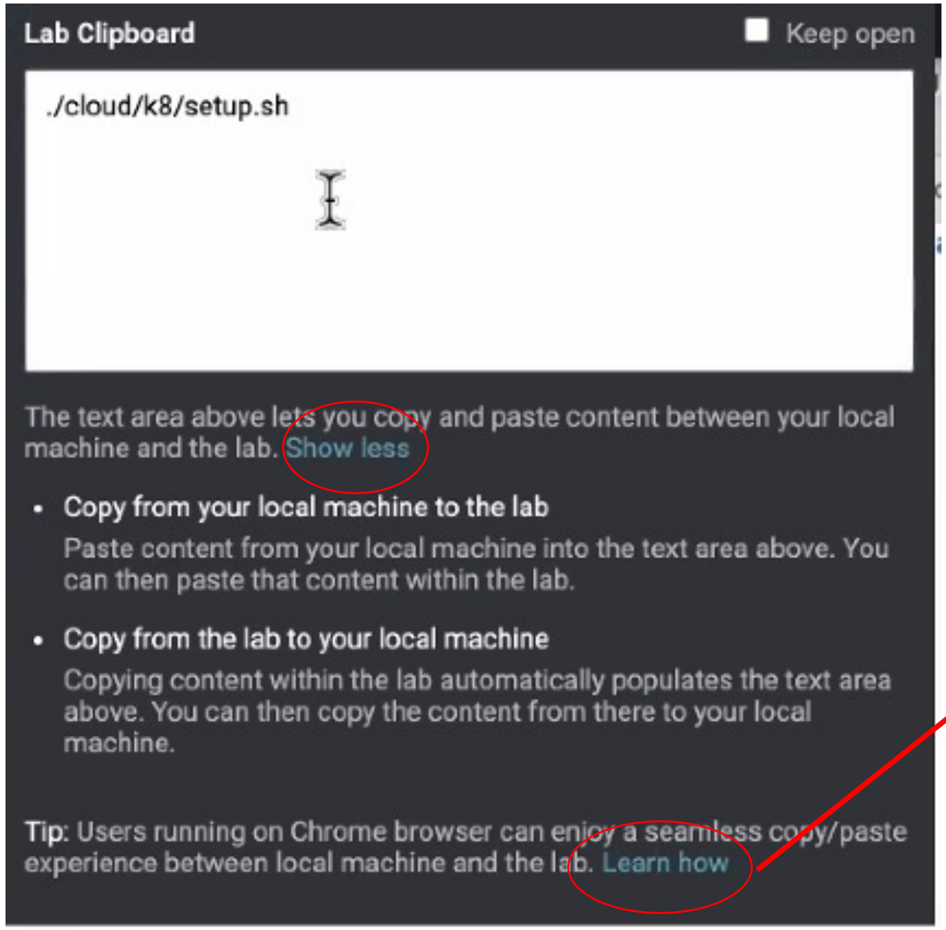
Follow instructions in

docs/workshops/sp1/01-SETUP.md



Allows Copy/Paste




Lab Clipboard

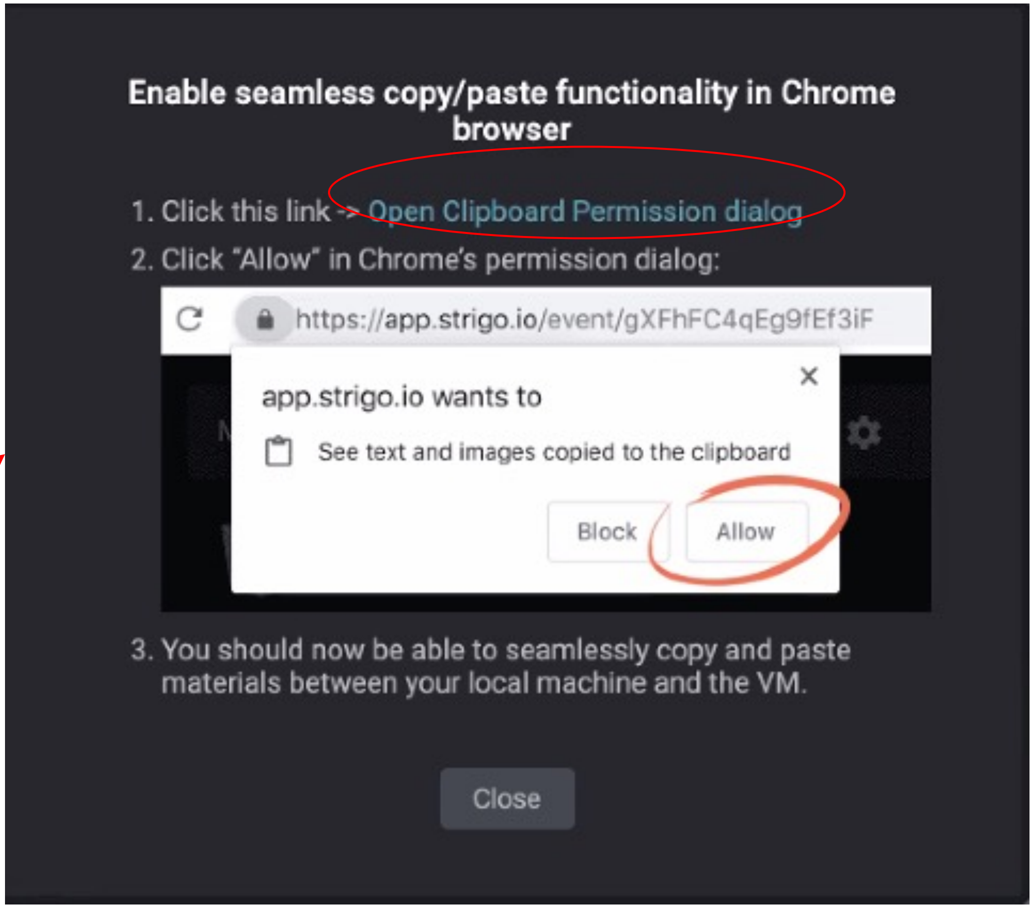
Keep open

```
./cloud/k8/setup.sh
```

The text area above lets you copy and paste content between your local machine and the lab. Show less

- Copy from your local machine to the lab
  Paste content from your local machine into the text area above. You can then paste that content within the lab.
- Copy from the lab to your local machine
  Copying content within the lab automatically populates the text area above. You can then copy the content from there to your local machine.

Tip: Users running on Chrome browser can enjoy a seamless copy/paste experience between local machine and the lab. Learn how


Enable seamless copy/paste functionality in Chrome browser

1. Click this link -> Open Clipboard Permission dialog
2. Click "Allow" in Chrome's permission dialog:

https://app.strigo.io/event/gXFhFC4qEg9fEf3iF

app.strigo.io wants to

See text and images copied to the clipboard

Block Allow

3. You should now be able to seamlessly copy and paste materials between your local machine and the VM.

Close

VMware Tanzu – Data Services

VMware Tanzu

Infrastructure for running modern apps and backing services with consistent, conformant Kubernetes everywhere.

Data Management
Management for Tanzu Data Services instances

GemFire
Fast In-Memory data store for Caching, Transactional and NoSQL support powered by Apache Geode

I need a fast data store

SQL
Relational MySQL or Postgres database for Transactional or Analytic data processing

I need to replatform a relational database

Greenplum
Massively Parallel Processing (MPP) Postgres for Big Data store for analytics, Machine Learning and Artificial Intelligence

I need to drive analytic value of out tons of existing data

Rabbit MQ
High throughput broker for reliable messaging delivery

I need reliable messaging delivery

Spring Cloud Data Flow
Data integration orchestration service for dynamically building data pipelines

I need flexible and manageable data integrations

Features

- ✓ Cloud deployed backing-services
- ✓ On-Premise and Multi-Cloud
- ✓ Self – Service
- ✓ Scaling
- ✓ HA - Fault Tolerant
- ✓ Based on open source
- ✓ World Class Support

RabbitMQ – 101 – Broker, Producers & Consumers

RabbitMQ is a message broker

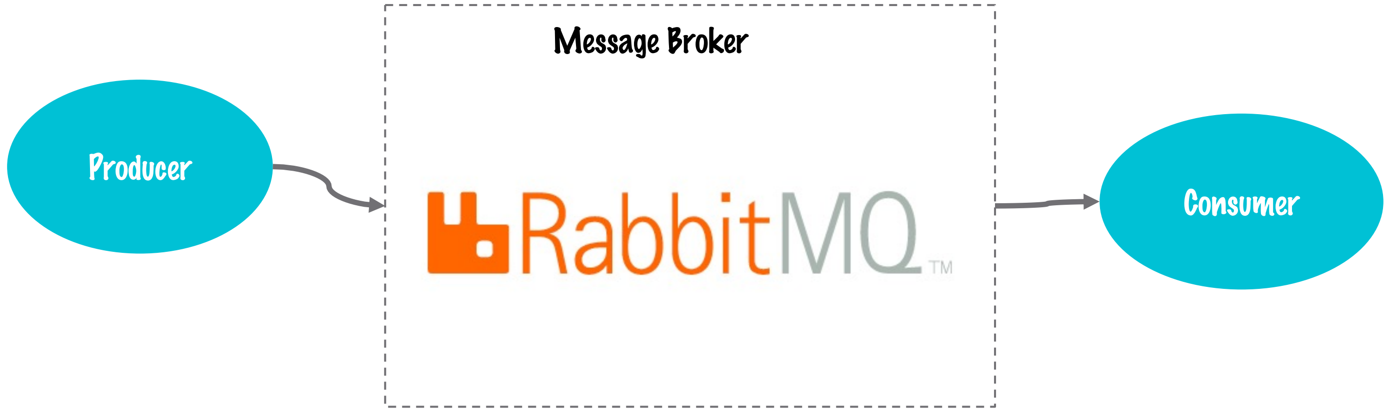
- stores and forwards binary blobs of data – messages.

Producer

- Program that sends messages is a producer

Consumer

- Program that mostly waits to receive messages:



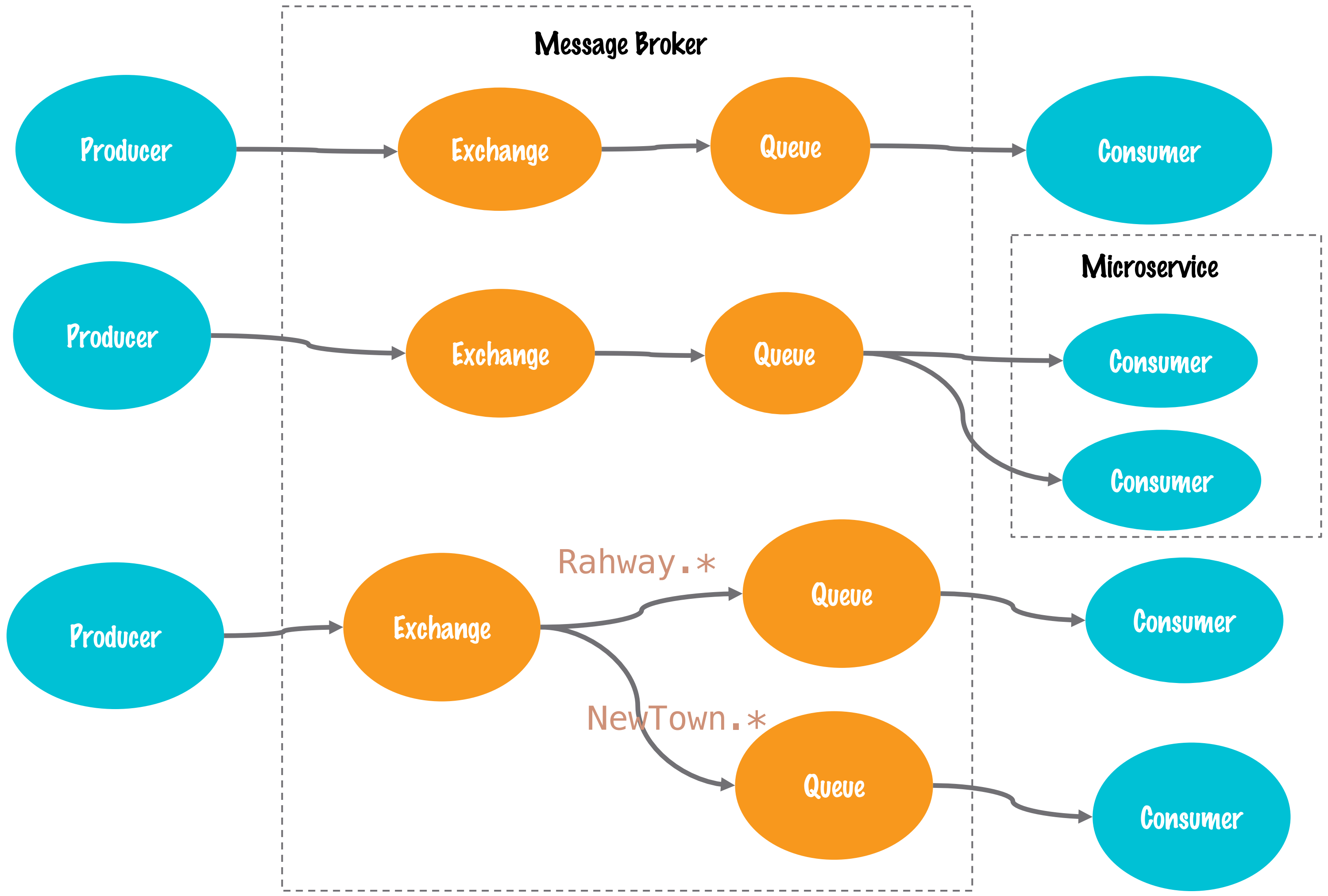
RabbitMQ – Exchanges & Queues

Queue

- Storage destination of messages inside RabbitMQ

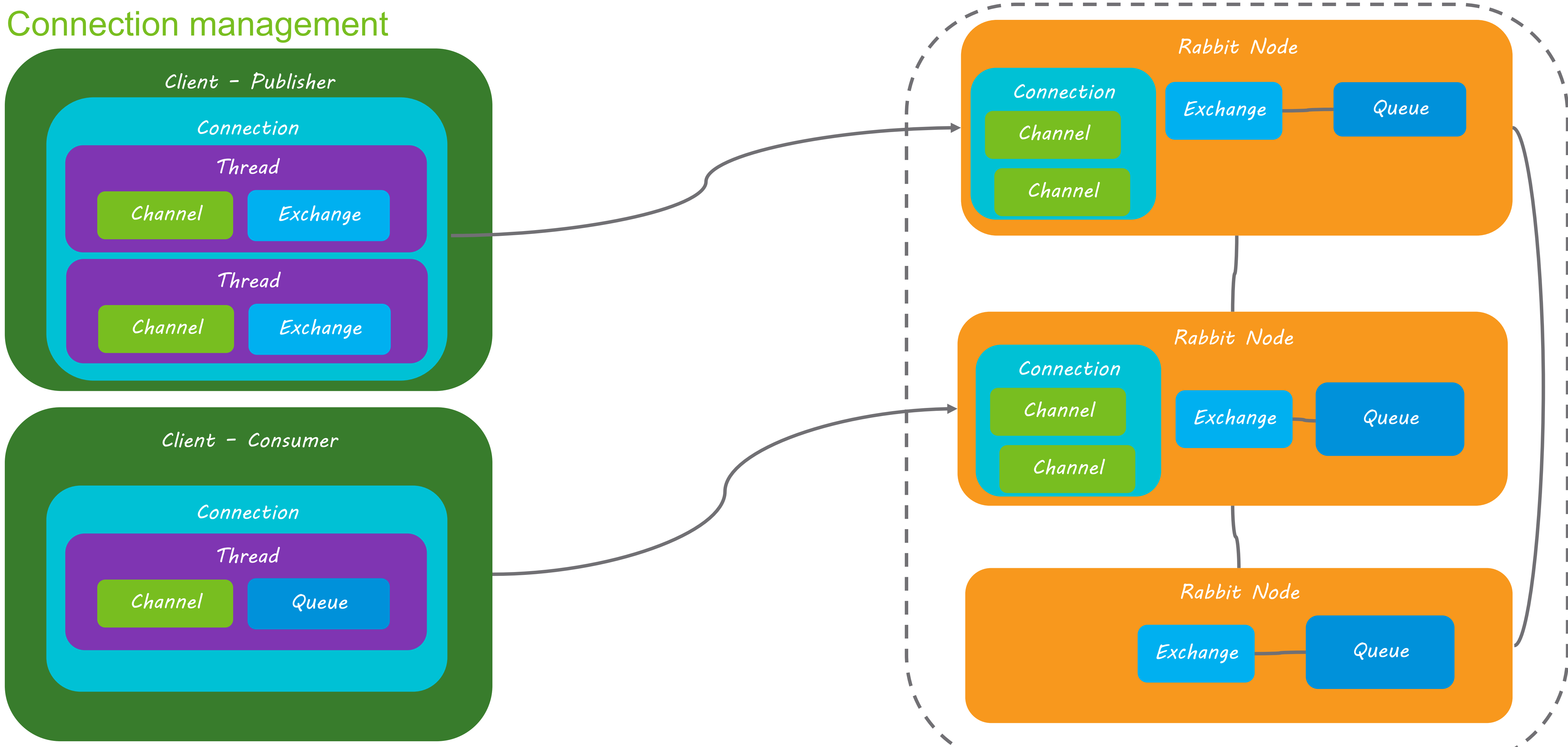
Exchanges

- Messages always sent to exchanges, then forwarded to queues based on routing rules.



RabbitMQ Clustering

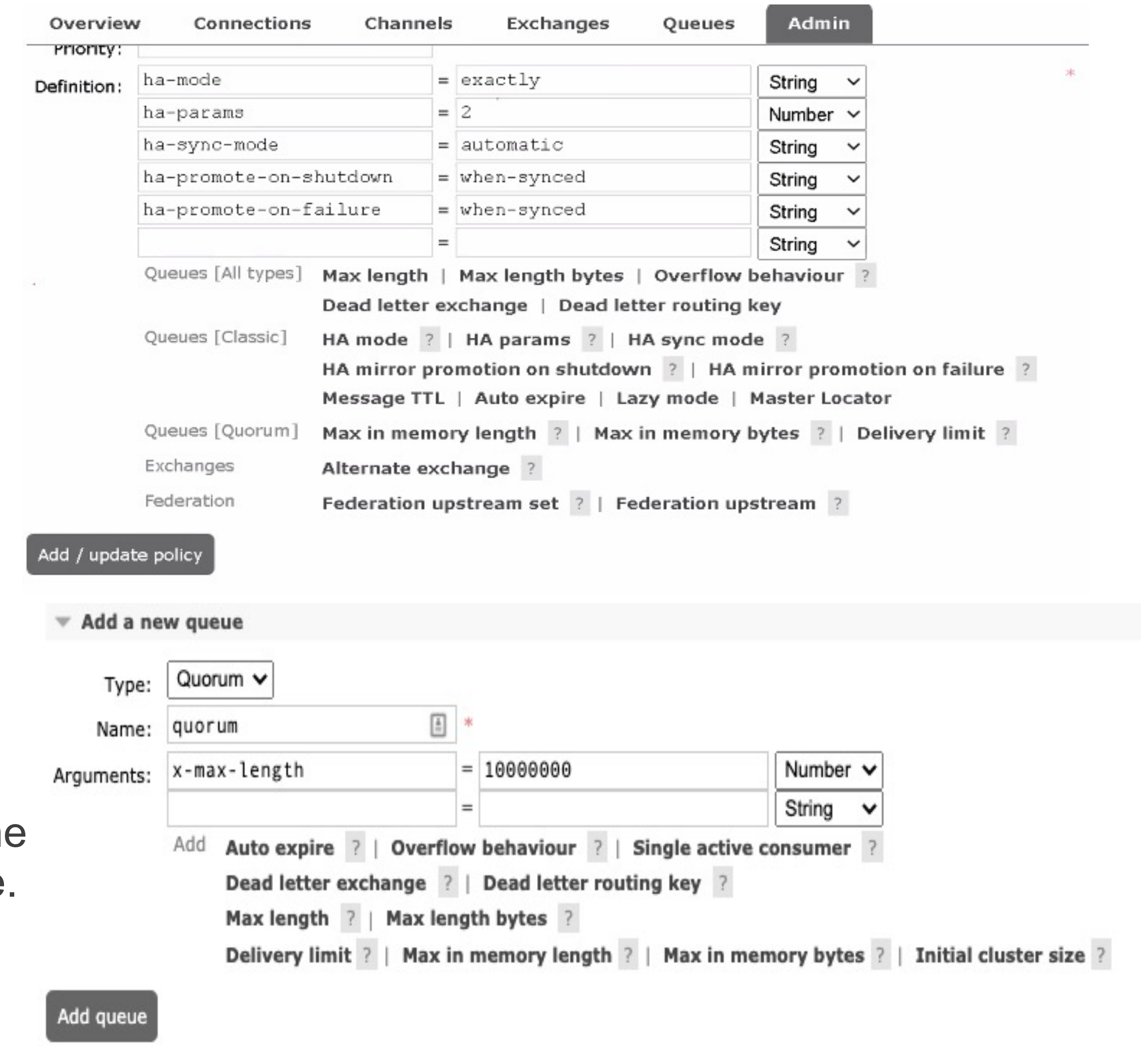
Connection management



Queues

Classic versus Quorum

- Classic Queues
  - Supports In-Memory messages
  - Option durable and or persisted messages
  - Mirrored replication through policy (deprecated)
- Quorum Queues
  - A durable, replicated with persisted messages
  - Based on the Raft consensus algorithm.
  - Preferred queue type over durable mirrored classics queues.
  - Quorum queues should be considered the default option for a replicated queue type.



Priority:

Definition:

ha-mode	=	exactly	String
ha-params	=	2	Number
ha-sync-mode	=	automatic	String
ha-promote-on-shutdown	=	when-synced	String
ha-promote-on-failure	=	when-synced	String
	=		String

Queues [All types]	Max length | Max length bytes | Overflow behaviour ? Dead letter exchange | Dead letter routing key
Queues [Classic]	HA mode ? | HA params ? | HA sync mode ? HA mirror promotion on shutdown ? | HA mirror promotion on failure ? Message TTL | Auto expire | Lazy mode | Master Locator
Queues [Quorum]	Max in memory length ? | Max in memory bytes ? | Delivery limit ?
Exchanges	Alternate exchange ?
Federation	Federation upstream set ? | Federation upstream ?

Add / update policy

Add a new queue

Type: Quorum

Name: quorum

Arguments:

x-max-length	=	10000000	Number
	=		String

Add Auto expire ? | Overflow behaviour ? | Single active consumer ?
Dead letter exchange ? | Dead letter routing key ?
Max length ? | Max length bytes ?
Delivery limit ? | Max in memory length ? | Max in memory bytes ? | Initial cluster size ?

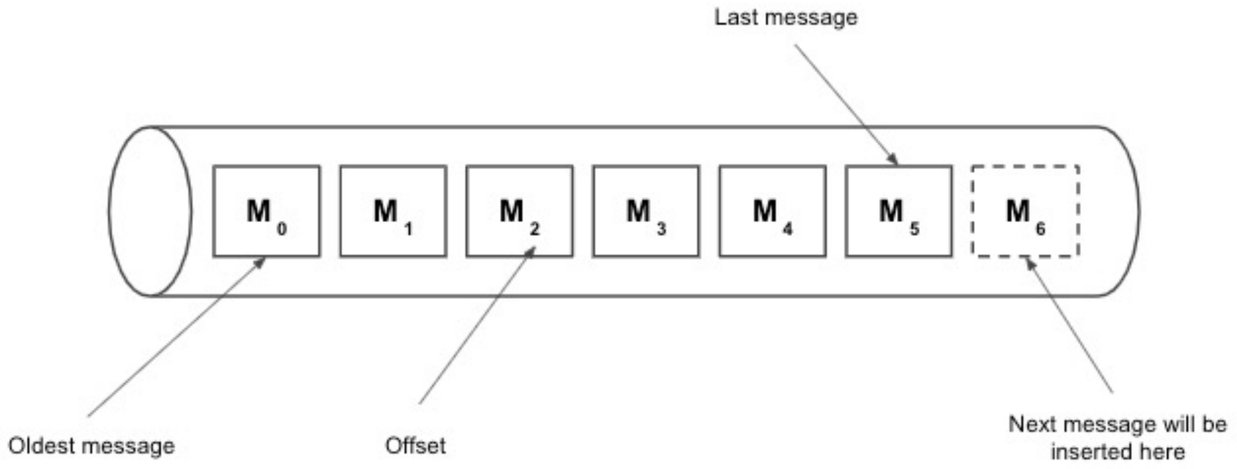
Add queue

Messaging Streaming - Queue

Replay Messages

- Kafka like event logging
- **Large fan-outs:** when several consumer applications need to read the same messages.
- **Replay / Time-traveling:** when consumer applications need to read the whole history of data or from a given point in a stream.
- **Throughput performance:** when higher throughput than with other protocols (AMQP, STOMP, MQTT) is required.
- **Large logs:** when large amount of data need to be stored, with minimal in-memory overhead.

```
channel.queueDeclare(
  "my-stream",
  true,           // durable
  false, false, // not exclusive, not auto-delete
  Collections.singletonMap("x-queue-type", "stream")
);
```

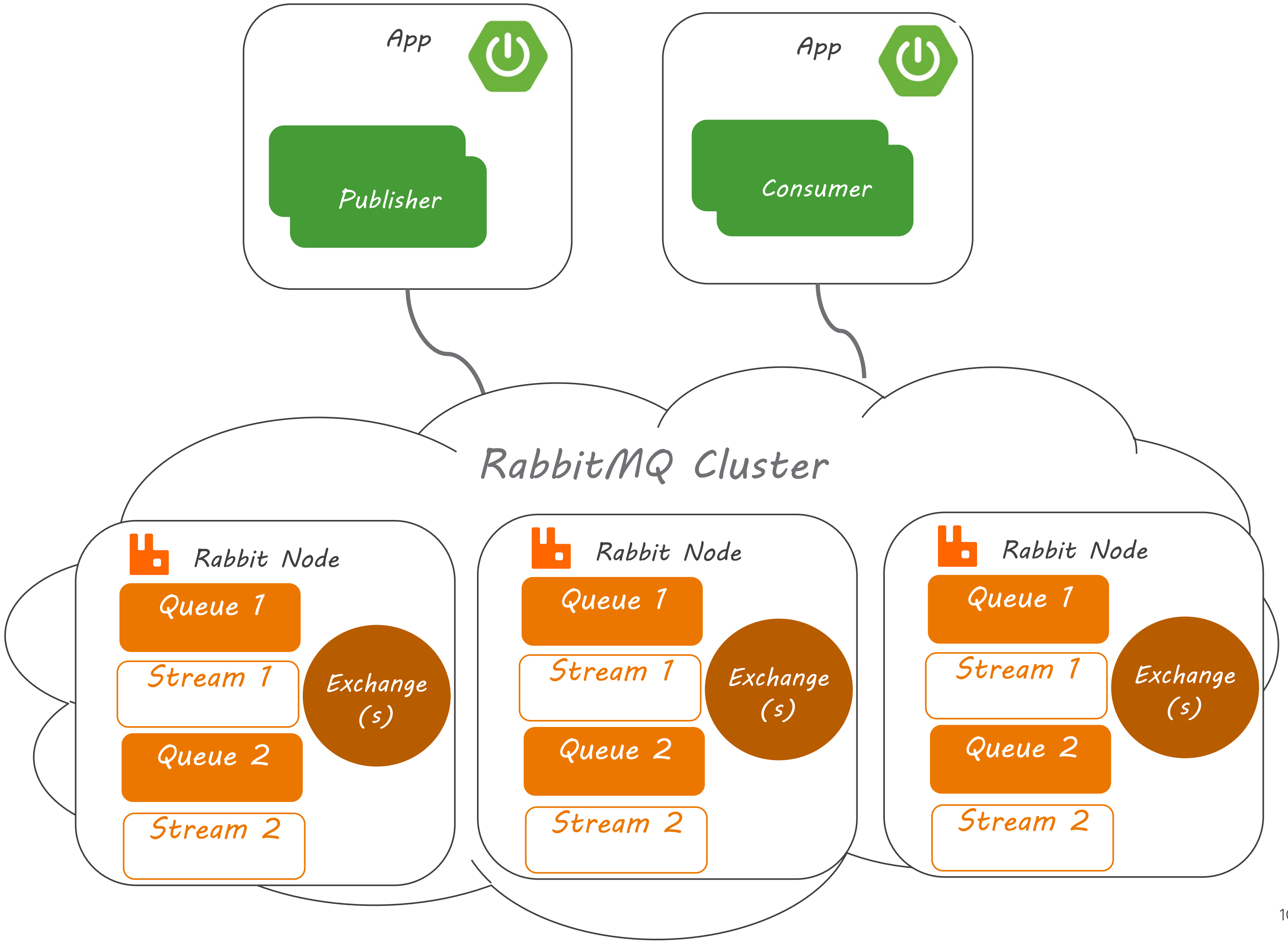


```
channel.basicConsume(
  "my-stream",
  false, // not auto-ack
  Collections.singletonMap("x-stream-offset", 0),
  (s, delivery) -> { }, // delivery callback
  s -> { } // cancel callback
);
```

RabbitMQ

Scalability/Reliability

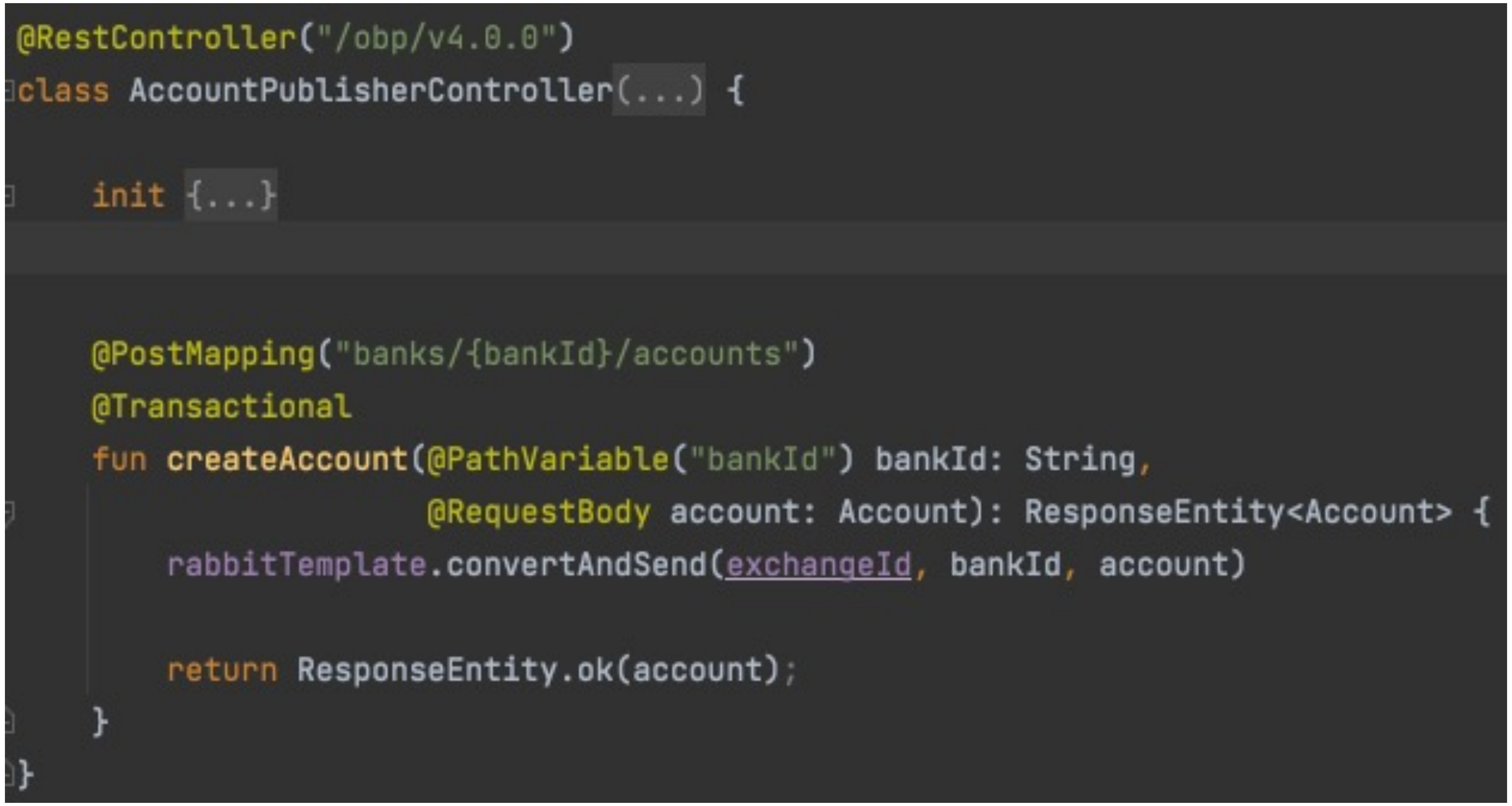
- An odd number of cluster nodes are recommended (ex: 1, 3, 5, 7) by several features that require a consensus between cluster member
- A client can connect to any node.
- Nodes will route operations to the leader.
- Add more queues for Scability across the cluster



Spring AMPQ

Publisher

- Define a Topic Exchange bean for automatic exchange creation
- RabbitTemplate can be used for sending messages
- Use @Transaction for Publisher confirms

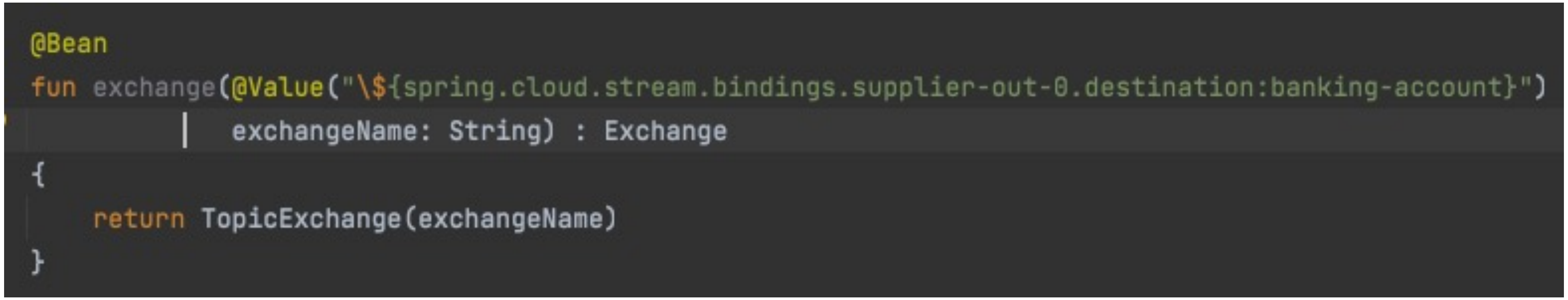


```
@RestController("/obp/v4.0.0")
class AccountPublisherController(...) {

    init {...}


    @PostMapping("banks/{bankId}/accounts")
    @Transactional
    fun createAccount(@PathVariable("bankId") bankId: String,
                      @RequestBody account: Account): ResponseEntity<Account> {
        rabbitTemplate.convertAndSend(exchangeId, bankId, account)

        return ResponseEntity.ok(account);
    }
}
```



```
@Bean
fun exchange(@Value("\${spring.cloud.stream.bindings.supplier-out-0.destination:banking-account}")
             exchangeName: String) : Exchange
{
    return TopicExchange(exchangeName)
}
```

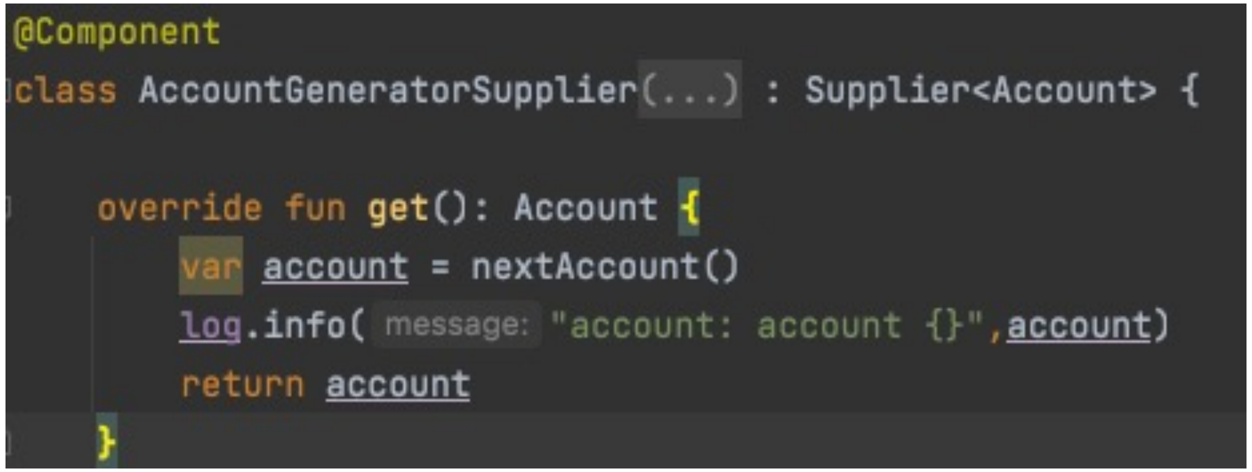
Spring Cloud Stream

Publishers

- Publisher
  implement java.util.function.Supplier

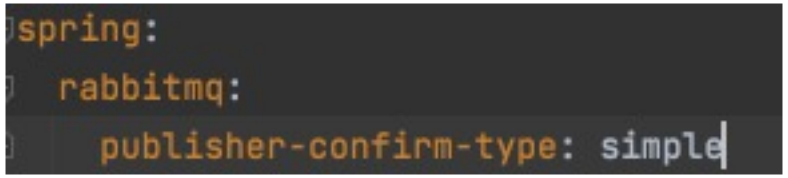
- spring.rabbitmq.publisher-confirm-type
  - SIMPLE
    - Use RabbitTemplate#waitForConfirms() (or wait ForConfirmsOrDie() within scoped operations.

- Support for RabbitMQ Streams
  - Spring Boot 2.6.0-M2
    - spring-rabbit-stream
    - spring-cloud-stream-binder-rabbit

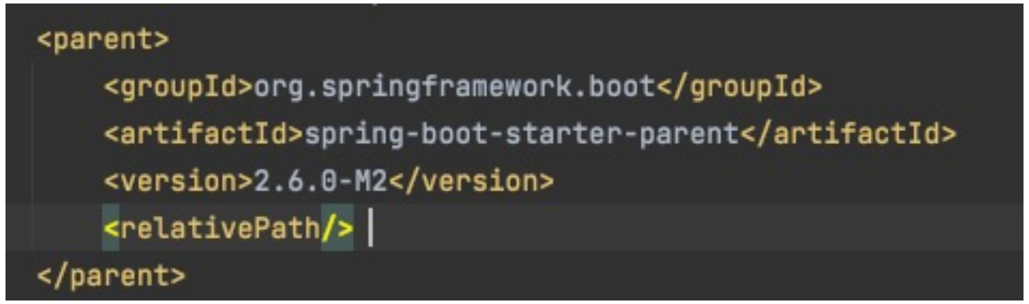


```
@Component
class AccountGeneratorSupplier(...) : Supplier<Account> {

    override fun get(): Account {
        var account = nextAccount()
        log.info( message: "account: account {}",account)
        return account
    }
```



```
spring:
  rabbitmq:
    publisher-confirm-type: simple
```

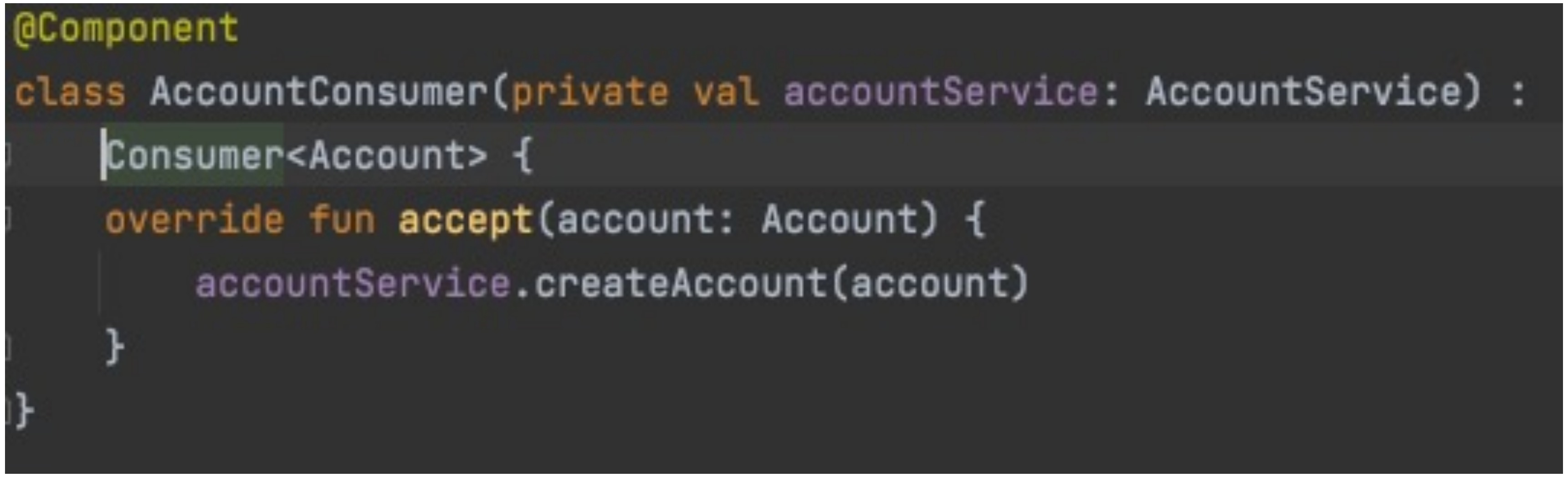


```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.6.0-M2</version>
    <relativePath/>
</parent>
```

Spring Cloud Stream

Consumers

- Publisher implement java.util.function.Consumer
- Default AcknowledgeMode = AUTO
  - Auto - the container will issue the ack/nack based on whether the listener returns normally, or throws an exception.
    - spring.cloud.stream.rabbit.bindings.<channelName>.consumer..



```
@Component
class AccountConsumer(private val accountService: AccountService) :
    Consumer<Account> {
    override fun accept(account: Account) {
        accountService.createAccount(account)
    }
}
```

Spring Cloud Data Flow

Build, Deploy, and Monitor streaming and batch data pipelines

- Spring Cloud Data Flow for VMware Tanzu automates the deployment of data pipelines backed by cloud native applications

Spring Cloud Stream

- Spring Cloud Stream is a framework for building highly scalable event-driven microservices connected with shared messaging systems.

Dashboard

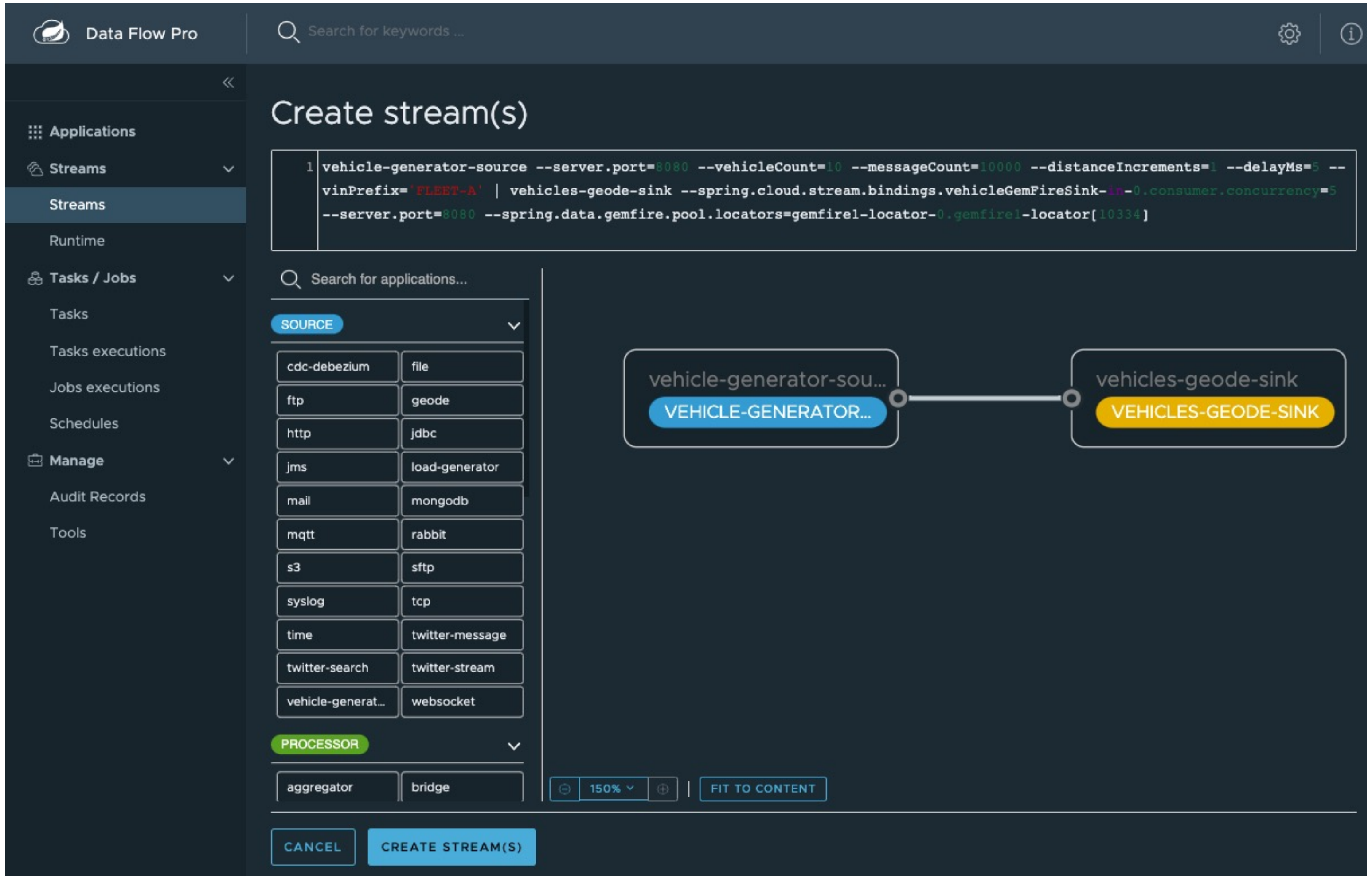
- GUI for managing data pipelines

DSL

- Pipeline definitions language similar to UNIX commands
  - **Ex: file | s3**

REST API & shell interface

```
dataflow:>stream list
```



Create stream(s)

```
vehicle-generator-source --server.port=8080 --vehicleCount=10 --messageCount=10000 --distanceIncrements=1 --delayMs=5 --
vinPrefix='FLEET-A' | vehicles-geode-sink --spring.cloud.stream.bindings.vehicleGemFireSink-in-0.consumer.concurrency=5
--server.port=8080 --spring.data.gemfire.pool.locators=gemfire1-locator-0.gemfire1-locator[10334]
```

Exercises

See https://github.com/Tanzu-Solutions-Engineering/tanzu-rabbitmq-event-streaming-showcase.git

Lab 1 - Setup RabbitMQ on K8
Lab 2 - Create a RabbitMQ Cluster with HA
Lab 3 - Spring Apps with Quorum Queues
Lab 4 - Spring Apps with Streams
Lab 5 - Spring Cloud DataFlow
Lab 6 - Provision RabbitMQ Topology Operation
- Users, Permissions, Queues, Vhost, etc.