[// References 1](#_Toc1553953544)

[Prerequsites for Lab 3](#_Toc2033137002)

[// Create spring boot project from STS 3](#_Toc62241727)

[// Spring boot App 5](#_Toc998087302)

[// Spring boot app : CommandLineRunner with component 5](#_Toc1607795028)

[// SpringBoot App: Autowiring Demo 6](#_Toc1543380967)

[// Spring Boot App: Unit(Integration) test 7](#_Toc668789315)

[// Create camel project without spring boot 8](#_Toc1752060868)

[// Create spring boot project from STS 9](#_Toc829913215)

[// Spring Boot App: With camel dependencies (main-run-controller=true) 11](#_Toc1212816979)

[// Spring Boot App: With camel dependencies - with Thread.sleep() 11](#_Toc666575244)

[// Route to move a file instead of copy : noop=false 12](#_Toc438936436)

[// Route to log message : log component 13](#_Toc2146094745)

[// Route to invoke bean method repeatedly : timer bean and stream component 14](#_Toc454783578)

[// Route to connect with another Route in same camel context : direct component 16](#_Toc2094783756)

[//Docker Introduction 19](#_Toc1077463638)

[// Create camel spring boot rabbitmq project 20](#_Toc992292984)

[// Add RabbitMQConsumer 20](#_Toc1651532854)

[// Add RabbitMQProducer 20](#_Toc447113704)

[// http client route 20](#_Toc370903297)

[// Create HTTP Server Route 20](#_Toc746787060)

[//Create REST API Route 20](#_Toc1914433114)

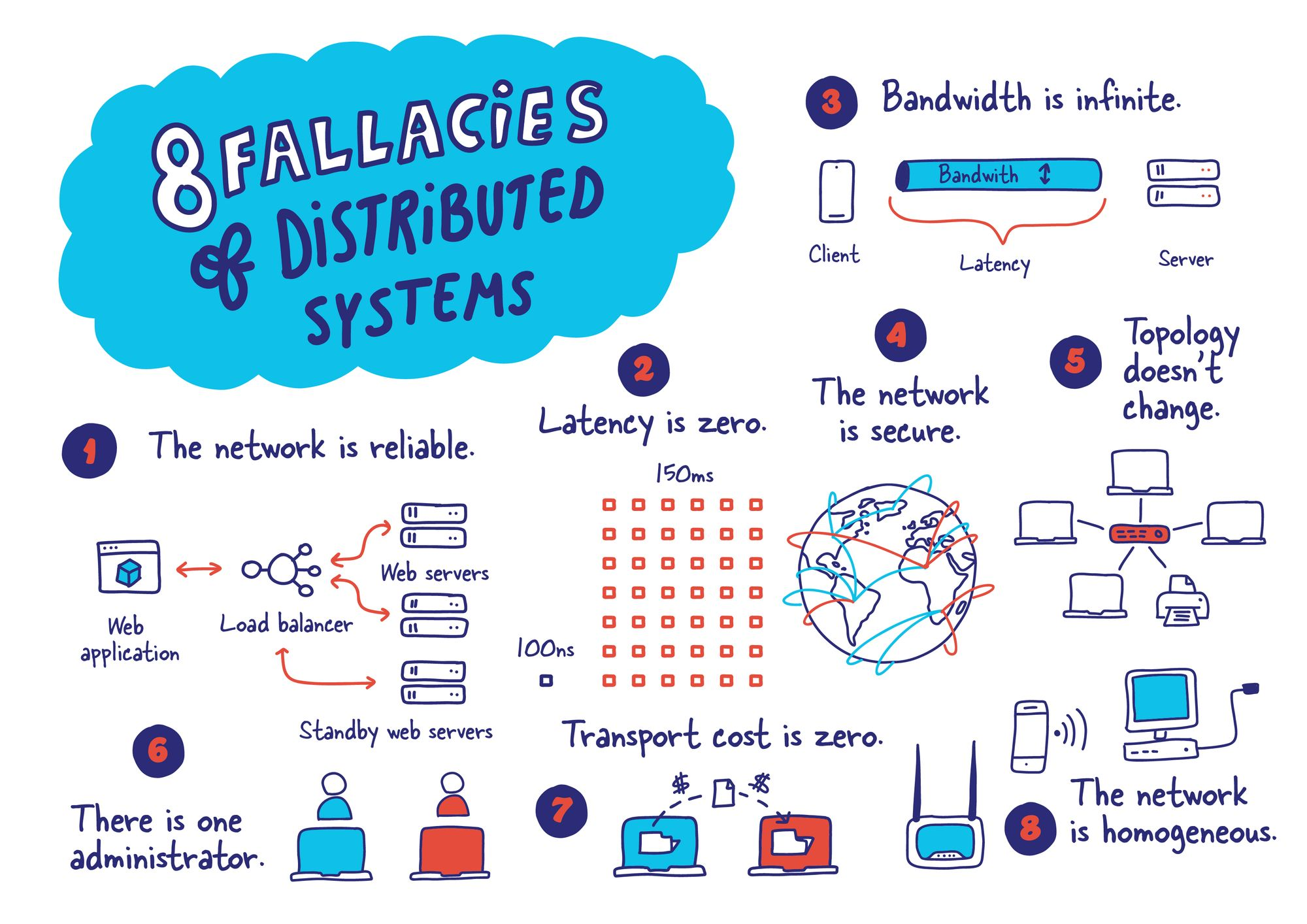
[// Bean Processor 20](#_Toc1799533042)

[// Content based processing 20](#_Toc1055926983)

[// Using Custom processor and filter 20](#_Toc1209484242)

[// Kafka producer and subscriber 20](#_Toc1703803090)

# // References



<https://en.wikipedia.org/wiki/Distributed_computing>

<https://www.enterpriseintegrationpatterns.com/patterns/messaging/index.html>

<https://camel.apache.org/components/3.20.x/eips/enterprise-integration-patterns.html>

<https://en.wikipedia.org/wiki/Design_Patterns>

<https://start.spring.io/> -> create spring boot project

<https://www.youtube.com/watch?v=gFozhTXOx18>

<https://hub.docker.com/_/rabbitmq>

<https://www.rabbitmq.com/tutorials/amqp-concepts.html>

<https://github.com/apache/camel-examples/tree/main/examples>

<https://camel.apache.org/blog/2022/02/camel-textual-debug-eclipse-vscode/>

<https://camel.apache.org/manual/predicate.html>

# Prerequsites for Lab

Install jdk 17 <https://download.oracle.com/java/17/archive/jdk-17.0.6_windows-x64_bin.msi>

Install STS (spring tool suite) <https://spring.io/tools>

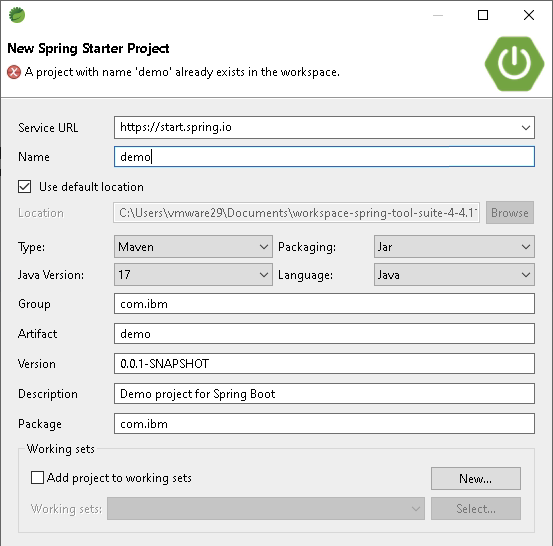
Install Docker Desktop (Check version & License) <https://www.docker.com/products/docker-desktop/>

Install VS Code <https://code.visualstudio.com/>

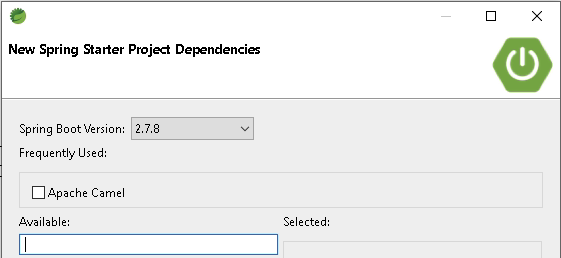
Install docker and kafka plugins

# // Create spring boot project from STS

* In STS choose new Spring starter Project



* Choose spring boot version do not select any dependencies



* Select default options and select finish

# // Spring boot App

package com.ibm;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

public class DemoApplication {

public static void main(String[] args) {

ConfigurableApplicationContext context = SpringApplication.run(DemoApplication.class, args);

System.out.println("Beans:" + context.getBeanDefinitionCount());

String[] beans = context.getBeanDefinitionNames();

for (String bean : beans) {

System.out.println(bean);

}

}

}

# // Spring boot app : CommandLineRunner with component

package com.ibm;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

public class DemoApplication implements CommandLineRunner{

public static void main(String[] args) {

ConfigurableApplicationContext context = SpringApplication.run(DemoApplication.class, args);

System.out.println("Beans:" + context.getBeanDefinitionCount());

}

@Override

public void run(String... args) throws Exception {

System.out.println("Inside commandline Runner");

}

}

package com.ibm;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

@Component

public class MyHello {

public MyHello() {

System.out.println("Hello");

}

@Value("${message}")

private String message;

public String getMessage() {

return message;

}

}

# // SpringBoot App: Autowiring Demo

package com.ibm;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ConfigurableApplicationContext;

@SpringBootApplication

public class DemoApplication implements CommandLineRunner{

@Autowired

private MyHello hello;

public static void main(String[] args) {

System.out.println("start");

ConfigurableApplicationContext context = SpringApplication.run(DemoApplication.class, args);

System.out.println("Beans:" + context.getBeanDefinitionCount());

}

@Override

public void run(String... args) throws Exception {

System.out.println("Inside commandline Runner");

System.out.println(hello.getMessage());

}

}

# // Spring Boot App: Unit(Integration) test

package com.ibm;

import static org.junit.jupiter.api.Assertions.assertTrue;

import org.junit.jupiter.api.Test;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

class DemoApplicationTests {

@Autowired

private MyHello hello;

@Test

void contextLoads() {

assertTrue(hello.getMessage().contains("Hello"));

}

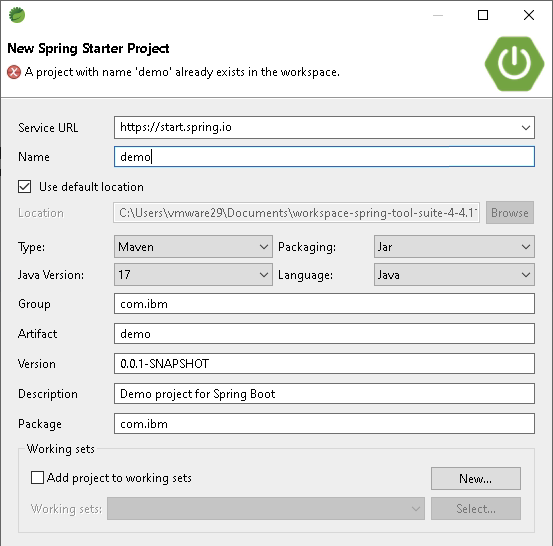
}

# // Create camel project without spring boot

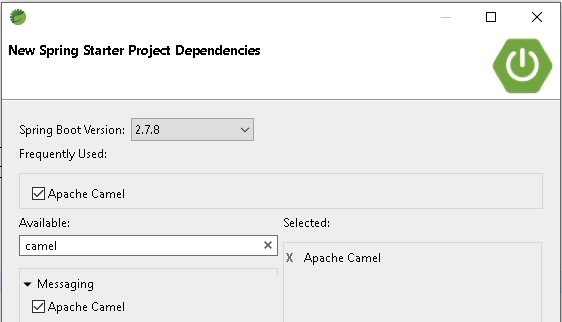
* In STS choose new Maven Project -> filter
* Enter camel-archetype-java
* Use default options

# // Create spring boot project from STS

* In STS choose new Spring starter Project



* Choose spring boot version and select camel dependency



* Select default options

# // Spring Boot App: With camel dependencies (main-run-controller=true)

import org.apache.camel.builder.RouteBuilder;

@Component

public class HelloRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("file:data/input?noop=true").to("file:data/output");

}

}

* Add camel.springboot.main-run-controller=true to application. Properties to run spring boot forever(untill its stopped)
* Create sample file in data/input and add some content
* Running above code should copy that file from input to output

# // Spring Boot App: With camel dependencies - with Thread.sleep()

@SpringBootApplication

public class CamelbootApplication {

public static void main(String[] args) {

ConfigurableApplicationContext context = SpringApplication.run(CamelbootApplication.class, args);

System.out.println("Beans:" + context.getBeanDefinitionCount());

System.out.println("sleeping");

try {

Thread.sleep(5\_000);

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println("bye");

}

}

* Add camel.springboot.main-run-controller=**false** to application properties (this is to remove warning message from log)
* Create sample file in data/input directory
* Run above code and it should copy and the program should exit after 5 sec

# // Route to move a file instead of copy : noop=false

@Component

public class FileMoveRoute extends RouteBuilder {

@Override

public void configure() throws Exception {

from("file:data/input")

.to("file:data/output");

}

}

# // Route to log message : log component

@Component

public class LogRoute extends RouteBuilder {

@Override

public void configure() throws Exception {

from("file:data/input?noop=true")

.to("log:?showAll=true")

.to("file:data/output");

}

}

# // Route to invoke bean method repeatedly : timer bean and stream component

Add below to pom.xml to include stream component

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-stream-starter</artifactId>

<version>3.20.2</version>

</dependency>

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

import com.ibm.MyBean;

@Component

public class BeanRoute extends RouteBuilder {

@Autowired

private MyBean myBean;

@Override

public void configure() throws Exception {

from("timer:hello1?period=1000&repeatCount=5")

// name the route

.routeId("hello")

// and call the bean

.bean(myBean, "saySomething('MyBean')")

// and print it to system out via stream component

.to("stream:out");

}

}

package com.ibm;

import java.util.concurrent.atomic.AtomicInteger;

import org.springframework.stereotype.Component;

@Component

public class MyBean {

private AtomicInteger counter = new AtomicInteger();

public String saySomething(String body) {

var count = counter.incrementAndGet();

var tname = Thread.currentThread().getName();

return String.format("%10s %s I am invoked %d times", tname, body, count);

}

}

# // Route to connect with another Route in same camel context : direct component

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

import com.ibm.MyBean;

@Component

public class TimerRoute extends RouteBuilder {

@Autowired

private MyBean myBean;

@Override

public void configure() throws Exception {

from("timer:hello2?period=1000&repeatCount=5")

.routeId("timer-route")

.bean(myBean, "saySomething('IBM1')")

.to("stream:out")

.to("direct:start-greetings");

//below can be defined here or in its own Component

//gets invoked in the same camel context (same thread) synchronously

// from("direct:start-greetings")

// .routeId("greetings-route")

// .bean(myBean, "saySomething('IBM2')")

// .to("stream:out");

}

}

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class DirectGreetingsRoute extends RouteBuilder {

@Override

public void configure() throws Exception {

from("direct:start-greetings")

.routeId("greetings-route")

.setBody(constant("Hello IBM!"))

.to("stream:out");

}

}

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class DirectLogRoute extends RouteBuilder {

@Override

public void configure() throws Exception {

from("direct:LogShowAll").to("log:?showAll=true");

}

}

# //Docker Introduction

C:\Users\vmware29>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

C:\Users\vmware29>docker run hello-world

Unable to find image 'hello-world:latest' locally

latest: Pulling from library/hello-world

2db29710123e: Pull complete

Digest: sha256:6e8b6f026e0b9c419ea0fd02d3905dd0952ad1feea67543f525c73a0a790fefb

Status: Downloaded newer image for hello-world:latest

Hello from Docker!

…

…

C:\Users\vmware29>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

C:\Users\vmware29>docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

d1e40499ba8f hello-world "/hello" 9 minutes ago Exited (0) 9 minutes ago objective\_wu

// Run RabbitMQ docker image

docker run -it -d --name rabbitmq -p 5672:5672 -p 15672:15672 rabbitmq:3.10-[management-alpine](https://github.com/docker-library/rabbitmq/blob/6e226fe8e99702c8726d5e7d5c5864e69548048d/3.10/alpine/management/Dockerfile)

Go to <http://localhost:15672/>

Username: guest

Password: guest

Create a queue name: tasks

Create an exchange : tasks-exchange

Create a binding : tasks

# // Create camel spring boot rabbitmq project

And add below dependency

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-spring-boot-starter</artifactId>

<version>3.20.2</version>

</dependency>

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-rabbitmq-starter</artifactId>

<version>3.20.2</version>

</dependency>

# // Add RabbitMQConsumer

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class RabbitMQConsumerRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("rabbitmq:mytasks-exchange?routingKey=tasks&queue=tasks&autoDelete=false").to("log:?showBody=true");

}

}

camel.springboot.main-run-controller=true

# // Add RabbitMQProducer

package com.ibm;

import org.apache.camel.Endpoint;

import org.apache.camel.Produce;

import org.apache.camel.ProducerTemplate;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class RabbitmqcamelbootApplication implements CommandLineRunner{

@Autowired

@Produce("direct:publishToRabbitMQ")

private ProducerTemplate template;

public static void main(String[] args) {

SpringApplication.run(RabbitmqcamelbootApplication.class, args);

try {

Thread.sleep(2000);

} catch (InterruptedException e) {

e.printStackTrace();

}

System.out.println("Published messages to queue");

}

@Override

public void run(String... args) throws Exception {

Endpoint endpoint=template.getDefaultEndpoint();

template.asyncSendBody(endpoint, "Hello World");

}

}

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class RabbitMQDirectProducerRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("direct:publishToRabbitMQ")

.routeId("publishToRabbitMQ")

.to("rabbitmq:mytasks-exchange?routingKey=tasks&queue=tasks&autoDelete=false").end();

}

}

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class RabbitMQProducerRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("timer:timer1?period=100&repeatCount=5")

.routeId("timely\_publisher")

.setBody(simple("Message"))

.to("log:?showAll=true")

.to("rabbitmq:mytasks-exchange?routingKey=tasks&queue=tasks&autoDelete=false");

}

}

# // http client route

Create a new spring starter project with camel and add below additional dependencies

camel.springboot.main-run-controller=true

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-spring-boot-starter</artifactId>

<version>3.20.2</version>

</dependency>

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-http-starter</artifactId>

<version>3.20.2</version>

</dependency>

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-jackson-starter</artifactId>

<version>3.20.2</version>

</dependency>

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-stream-starter</artifactId>

<version>3.20.2</version>

</dependency>

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-jetty-starter</artifactId>

<version>3.20.2</version>

</dependency>

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class HttpClientRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("timer:hello?period=5000&repeatCount=1")

.routeId("HTTPBINCLIENT")

//.setHeader

.to("http://www.httpbin.org/get?name=Siva")

.to("stream:out");

}

}

# // Create HTTP Server Route

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.apache.camel.model.rest.RestBindingMode;

import org.springframework.stereotype.Component;

@Component

public class HttpServerRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

restConfiguration().component("jetty").host("localhost").port(8080).bindingMode(RestBindingMode.json);

from("rest:get:hello")

.transform().constant("Bye World");

from("rest:get:hello:/{me}")

.transform().simple("Hi ${header.me}");

from("rest:get:hello:/french/{me}")

.transform().simple("Bonjour ${header.me}");

}

}

http://localhost:8080/hello/siva

http://localhost:8080/hello/french/siva

# //Create REST API Route

package com.ibm.routes;

import java.util.Map;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class HttpRestApiRoute extends RouteBuilder {

@Override

public void configure() throws Exception {

rest("/say")

.get("/hello").to("direct:hello")

.get("/bye").consumes("application/json").to("direct:bye")

.post("/bye").to("mock:update");

from("direct:hello")

.transform().constant("Hello World");

from("direct:bye")

.transform().constant(Map.of("key","Bye World"));

}

}

<http://localhost:8080/say/hello>

<http://localhost:8080/say/bye>

# // Bean Processor

Create new spring starter project with camel and add below dependencies

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-stream-starter</artifactId>

<version>3.20.2</version>

</dependency>

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-gson-starter</artifactId>

<version>3.20.2</version>

</dependency>

package com.ibm.domain;

public class Employee {

private int empid;

private String name;

private int age;

public int getEmpid() {

return empid;

}

public void setEmpid(int empid) {

this.empid = empid;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public Employee getEmployee() {

return this;

}

public Employee() {

// TODO Auto-generated constructor stub

}

public Employee(int empid, String name, int age) {

super();

this.empid = empid;

this.name = name;

this.age = age;

}

@Override

public String toString() {

return "Employee [empid=" + empid + ", name=" + name + ", age=" + age + "]";

}

}

//Create REST API Route

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.apache.camel.model.dataformat.JsonLibrary;

import org.springframework.stereotype.Component;

import com.ibm.domain.Employee;

@Component

public class ProcessorRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("timer:hello?repeatCount=1")

.bean(new Employee(1,"Siva",32), "getEmployee()")

.marshal()

.json(JsonLibrary.Gson, true)

.to("file://data/output?fileName=employee.json");

}

}

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

import com.ibm.domain.Employee;

@Component

public class BeanProcessorRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("timer:hello?repeatCount=1")

.bean(new Employee(1,"Siva",32), "getEmployee()")

.to("stream:out");

}

}

# // Content based processing

Create new spring starter project with camel dependency

package com.ibm.routes;

import static org.apache.camel.support.builder.PredicateBuilder.or;

import org.apache.camel.builder.RouteBuilder;

import org.apache.camel.builder.ValueBuilder;

import org.springframework.stereotype.Component;

@Component

public class CBRRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

ValueBuilder file = header("CamelFileName");

from("file:data/input1?noop=true")

.choice()

.when(simple("${in.headers.CamelFileName} ends with '.txt'"))

.to("file:data/output/text")

.when(or(file.endsWith("jpg"), file.endsWith("jpeg")))

.to("file:data/output/images")

.otherwise()

.to("file:data/output/ignore");

}

}

Copy text files and images to input1 directory and observe that files are copied based on extension

# // Using Custom processor and filter

package com.ibm.routes;

import org.apache.camel.Predicate;

import org.apache.camel.Processor;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Component;

@Component

public class FileProcessorRoute extends RouteBuilder{

@Autowired

private Processor processor;

@Autowired

private Predicate myFilter;

@Override

public void configure() throws Exception {

from("file:data/input2?noop=true")

.filter(myFilter)

.process(processor)

.to("file:data/output");

}

}

package com.ibm.routes;

import org.apache.camel.Exchange;

import org.apache.camel.Processor;

import org.springframework.stereotype.Component;

@Component

public class ToUpperCaseProcessor implements Processor {

@Override

public void process(Exchange exchange) throws Exception {

String body = exchange.getIn().getBody(String.class);

String outBody = body.toUpperCase();

exchange.getMessage().setBody(outBody);

}

}

package com.ibm.routes;

import org.apache.camel.Exchange;

import org.apache.camel.Predicate;

import org.springframework.stereotype.Component;

@Component

public class FileFilterPredicate implements Predicate{

@Override

public boolean matches(Exchange exchange) {

// check content

return exchange.getIn().getHeader("CamelFileName").toString().endsWith(".txt");

}

}

Copy text files and images to input2 directory and observe that only text files are being processed

# // Kafka producer and subscriber

Create a file called docker-compose.yaml and add below content

version: '3'

services:

zookeeper:

image: wurstmeister/zookeeper

kafka:

image: wurstmeister/kafka

ports:

- "9092:9092"

environment:

KAFKA\_ADVERTISED\_HOST\_NAME: localhost

KAFKA\_ZOOKEEPER\_CONNECT: zookeeper:2181

Then run below command in a command prompt

docker compose up

// Kafka producer route

Create new spring starter project with camel dependency

Add below dependency

<dependency>

<groupId>org.apache.camel.springboot</groupId>

<artifactId>camel-kafka-starter</artifactId>

<version>3.20.2</version>

</dependency>

package com.ibm.routes;

import org.apache.camel.builder.RouteBuilder;

import org.apache.camel.component.kafka.KafkaConstants;

import org.springframework.stereotype.Component;

@Component

public class KafkaProducerRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

from("timer:hello?repeatCount=1")

.setBody(constant("Message from Camel")) // Message to send

.setHeader(KafkaConstants.KEY, constant("Camel")) // Key of the message

.to("kafka:hello?brokers=localhost:9092");

}

}

// Kafka Consumer route

package com.ibm;

import org.apache.camel.builder.RouteBuilder;

import org.springframework.stereotype.Component;

@Component

public class KafkaConsumerRoute extends RouteBuilder{

@Override

public void configure() throws Exception {

// TODO Auto-generated method stub

from("kafka:hello?brokers=localhost:9092&autoOffsetReset=earliest")

.log("Message received from Kafka : ${body}")

.log(" on the topic ${headers[kafka.TOPIC]}")

.log(" on the partition ${headers[kafka.PARTITION]}")

.log(" with the offset ${headers[kafka.OFFSET]}")

.log(" with the key ${headers[kafka.KEY]}");

}

}