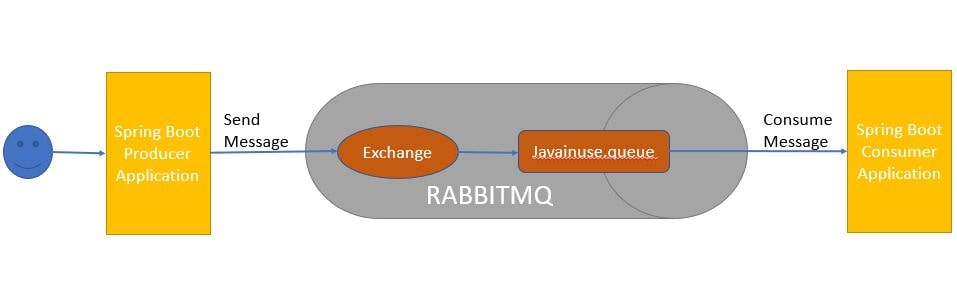
Spring Boot + RabbitMQ Consume Message Simple Example

In a [previous post we integrated RabbitMQ and Spring Boot to send a message to RabbitMQ.](https://www.javainuse.com/spring/spring-boot-rabbitmq-hello-world) In this post we consuming the message using Spring Boot and RabbitMQ.  
[In a previous post](https://www.javainuse.com/misc/rabbitmq-hello-world) we had seen how to get RabbitMQ up and running.  


RabbitMQ - Table Of Contents

[What is Messaging?](https://www.javainuse.com/messaging/rabbitmq/introtomessaging)

[Getting Started with RabbitMQ - Install and start RabbitMQ.](https://www.javainuse.com/misc/rabbitmq-hello-world)

[Spring Boot + RabbitMQ Publish Message Example](https://www.javainuse.com/spring/spring-boot-rabbitmq-hello-world)

[Spring Boot + RabbitMQ Tutorial - Configure Listeners to consume messages using MessageListenerContainer](https://www.javainuse.com/messaging/rabbitmq/listeners)

[**Spring Boot + RabbitMQ Consume Message Example using RabbitListener**](https://www.javainuse.com/spring/spring-boot-rabbitmq-consume)

[Spring Boot + RabbitMQ Tutorial - Implement Exchange Types](https://www.javainuse.com/messaging/rabbitmq/exchange)

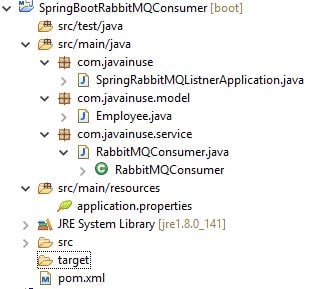
[Spring Boot + RabbitMQ Tutorial - Retry and Error Handling Example](https://www.javainuse.com/messaging/rabbitmq/error)

[Spring Cloud Stream - RabbitMQ Publish Message Example](https://www.javainuse.com/spring/cloud-stream-rabbitmq-1)

[Spring Cloud Stream - RabbitMQ Consume Message Example](https://www.javainuse.com/spring/cloud-stream-rabbitmq-2)

[Pivotal Cloud Foundry Tutorial - Deploying Spring Boot + RabbitMQ Application to PCF](https://www.javainuse.com/pcf/pcf-rabbitmq)

Lets Begin-

The project will be as follows-  
  
Define the pom.xml as follows- Add the **spring-boot-starter-amqp** dependency.

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.javainuse</groupId>

<artifactId>SpringBootRabbitMQConsumer</artifactId>

<version>0.0.1</version>

<packaging>jar</packaging>

<name>SpringBootRabbitMQConsumer</name>

<parent>

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

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

<version>1.5.4.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.json</groupId>

<artifactId>json</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

<description>SpringBootRabbitMQConsumer</description>

</project>

Define the domain class Employee as follows-

package com.javainuse.model;

import com.fasterxml.jackson.annotation.JsonIdentityInfo;

import com.fasterxml.jackson.annotation.ObjectIdGenerators;

@JsonIdentityInfo(generator = ObjectIdGenerators.IntSequenceGenerator.class, property = "@id", scope = Employee.class)

public class Employee {

private String empName;

private String empId;

public String getEmpName() {

return empName;

}

public void setEmpName(String empName) {

this.empName = empName;

}

public String getEmpId() {

return empId;

}

public void setEmpId(String empId) {

this.empId = empId;

}

@Override

public String toString() {

return "Employee [empName=" + empName + ", empId=" + empId + "]";

}

}

Define the RabbitMQConsumer class which consumes the message from RabbitMQ using RabbitListener.The RabbitMQ Listener listens to RabbitMQ Queue for any incoming messages. For the basic configuration we specify the the Queue/Topic Name (the name of the queue/topic where the message should be consumed)

package com.javainuse.service;

import org.springframework.amqp.rabbit.annotation.RabbitListener;

import org.springframework.stereotype.Component;

import com.javainuse.model.Employee;

@Component

public class RabbitMQConsumer {

@RabbitListener(queues = "${javainuse.rabbitmq.queue}")

public void recievedMessage(Employee employee) {

System.out.println("Recieved Message From RabbitMQ: " + employee);

}

}

Next define th following properties in application.properties-

spring.rabbitmq.host=localhost

spring.rabbitmq.port=5672

spring.rabbitmq.username=guest

spring.rabbitmq.password=guest

javainuse.rabbitmq.queue=javainuse.queue

server.port=8081

Finally Define the Spring Boot Class with @SpringBootApplication annotation

package com.javainuse;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringBootHelloWorldApplication {

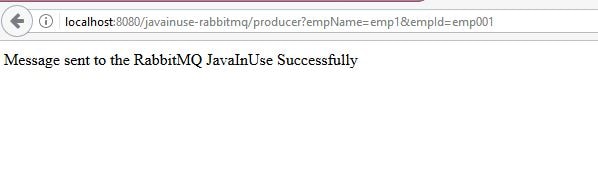
public static void main(String[] args) {

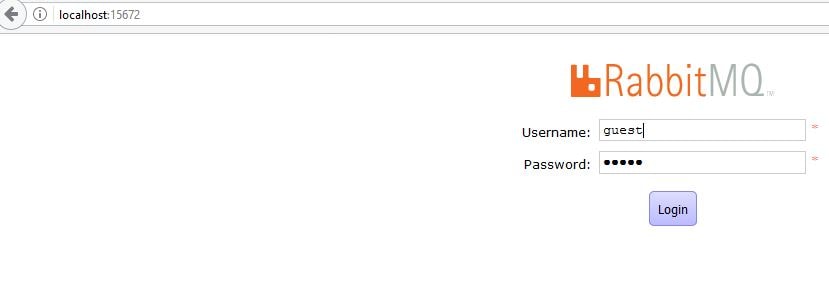
SpringApplication.run(

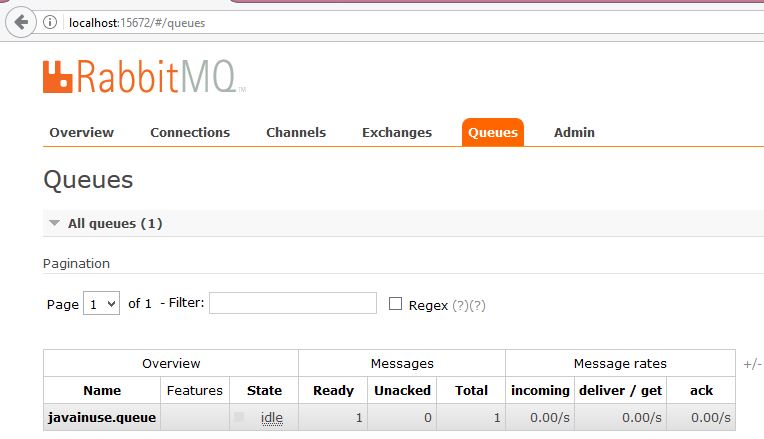
new Object[] { SpringBootHelloWorldApplication.class }, args);

}

}

We are done with the required Java code. Now lets start RabbitMQ. As we had explained in detail in the [Getting started with RabbitMQ](https://www.javainuse.com/misc/rabbitmq-hello-world) perform the steps to start the RabbitMQ.  
Next start the Spring Boot RabbitMQProducer application we developed in previous tutorial. Application by running it as a Java Application. Hit the url as follows- **http://localhost:8080/javainuse-rabbitmq/producer?empName=emp1&empId=emp001**  


This will trigger the message to be sent to the javainuse queue.  
Next go to the RabbitMQ console-**http://localhost:15672/**  


We can see in the Queues section, a queue name java gets created and it has one message.  


Next start the Spring Boot RabbitMQ Consumer application we just developed. It will consume the message from the queue named javainuse.queue.

