Национальный исследовательский технологический университет «МИСиС»
Институт Информационных технологий и компьютерных наук (ИТКН)

Курс «Системная и программная инженерия»

Лабораторная работа № 5
по теме
«Интеграция RabbitMQ с Java Spring»

Выполнил:
Смирнов А.А.

Проверил:
Козлов М.Е.

Москва, 2023

Цель: познакомиться со способом интеграции RabbitMQ с Java Spring. Выполнить задания:

1. Создать fanout обменник, в имени которого необходимо указать ФИО, и номер группы, создать очередь для него, связать их между собой, и отправить сообщение. Продемонстрировать отправку сообщения.
2. Создать direct обменник, в имени которого необходимо указать ФИО, и номер группы, создать очередь для него, связать их между собой, и отправить сообщение. Продемонстрировать отправку сообщения. (Подсказка: для реализации direct обменника необходимо использовать объект DirectExchange, а так же связывать его с очередью по ключу с помощью метода .with())

Ход работы:

Часть 1:

Для выполнения лабораторной работы я реализовал классы:

- Receiver отвечающий за получение сообщений

```
@Component
public class Receiver {

    public void receiveMessage(String message) {
        System.out.println("Получено сообщение: " + message);
    }
}
```

- Runner отвечающий за отправку сообщения

```
    private final RabbitTemplate rabbitTemplate;

    public Runner(RabbitTemplate rabbitTemplate) {
        this.rabbitTemplate = rabbitTemplate;
    }

    @Override
    public void run(String... args) throws Exception {
        while(true) {
            System.out.println("Sending message...");

rabbitTemplate.convertAndSend(RabbitMqLabaApplication.exchangeNa
me,
                    RabbitMqLabaApplication.queueName,
                    "Hello from " +
                    RabbitMqLabaApplication.exchangeName);
            Thread.sleep(2000);
        }
    }
```

- RabbitMqLabaApplication отвечающий за регистрацию fanout очередей и обменников и запуск приложения

```
    @SpringBootApplication
    public class RabbitMqLabaApplication {
        static final String rabbitHost = "localhost";
        static final String rabbitUser= "guest";
        static final String rabbitPassword = "guest";
        static final String exchangeName = "fanout-exchange-
smirnov-aa-bivt-20-1";
        static final String queueName = "fanout-queue-smirnov-aa-
bivt-20-1";

        @Bean
        public ConnectionFactory connectionFactory() {
            CachingConnectionFactory connectionFactory = new
CachingConnectionFactory();
            connectionFactory.setAddresses(rabbitHost);
            connectionFactory.setUsername(rabbitUser);
            connectionFactory.setPassword(rabbitPassword);
            return connectionFactory;
        }

        @Bean
        public RabbitTemplate rabbitTemplate(ConnectionFactory
connectionFactory) {
            final RabbitTemplate rabbitTemplate = new
RabbitTemplate(connectionFactory);
            return rabbitTemplate;
        }

        @Bean
        Queue queue() {
            return new Queue(queueName, false);
        }

        @Bean
        FanoutExchange exchange() {
            return new FanoutExchange(exchangeName);
        }

        @Bean
        Binding binding(Queue queue, FanoutExchange exchange) {
            return BindingBuilder.bind(queue).to(exchange);
        }

        @Bean
        SimpleMessageListenerContainer
container(ConnectionFactory connectionFactory,
```

```
MessageListenerAdapter listenerAdapter) {
            SimpleMessageListenerContainer container = new
SimpleMessageListenerContainer();
            container.setConnectionFactory(connectionFactory);
            container.setQueueNames(queueName);
            container.setMessageListener(listenerAdapter);
            return container;
        }

        @Bean
        MessageListenerAdapter listenerAdapter(Receiver
receiver) {
            return new MessageListenerAdapter(receiver,
"receiveMessage");
        }

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

```
Sending message...
Получено сообщение: Hello from fanout-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from fanout-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from fanout-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from fanout-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from fanout-exchange-smirnov-aa-bivt-20-1
```

Рисунок 1 – Прием сообщений.

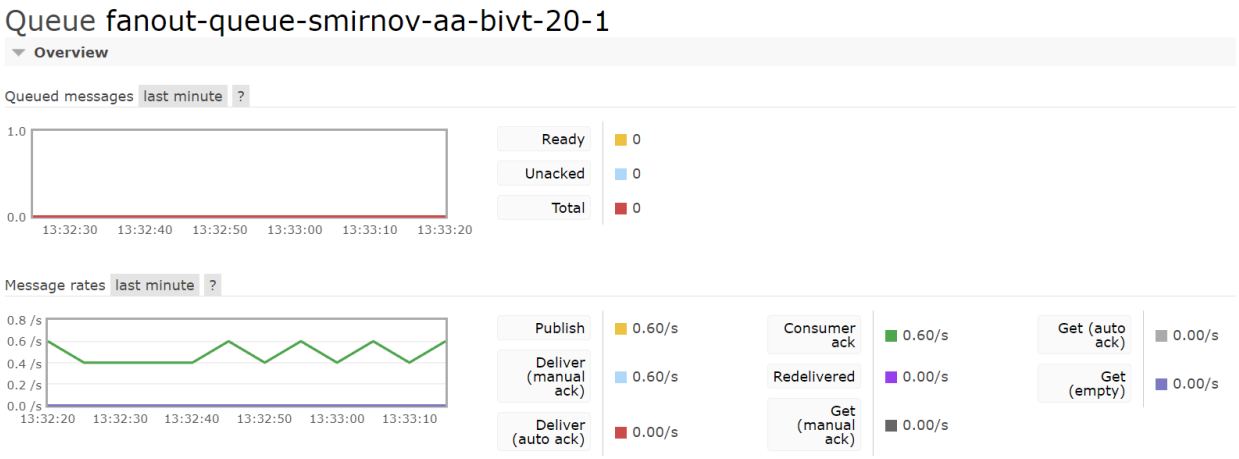

Рисунок 2 – Статистика fanout queue.

Часть 2:

Для создания direct очереди и обменника, я изменил переменные:

```
static final String exchangeName = "direct-exchange-smirnov-aa-bivt-20-1";
static final String queueName = "direct-queue-smirnov-aa-bivt-20-1";
```

Также изменил методы exchange и binding:

```
@Bean
DirectExchange exchange() {
    return new DirectExchange(exchangeName);
}

@Bean
Binding binding(Queue queue, DirectExchange exchange) {
    return BindingBuilder.bind(queue).to(exchange).withQueueName();
}
```

```
Sending message...
Получено сообщение: Hello from direct-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from direct-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from direct-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from direct-exchange-smirnov-aa-bivt-20-1
Sending message...
Получено сообщение: Hello from direct-exchange-smirnov-aa-bivt-20-1
```

Рисунок 3 – Прием сообщений.

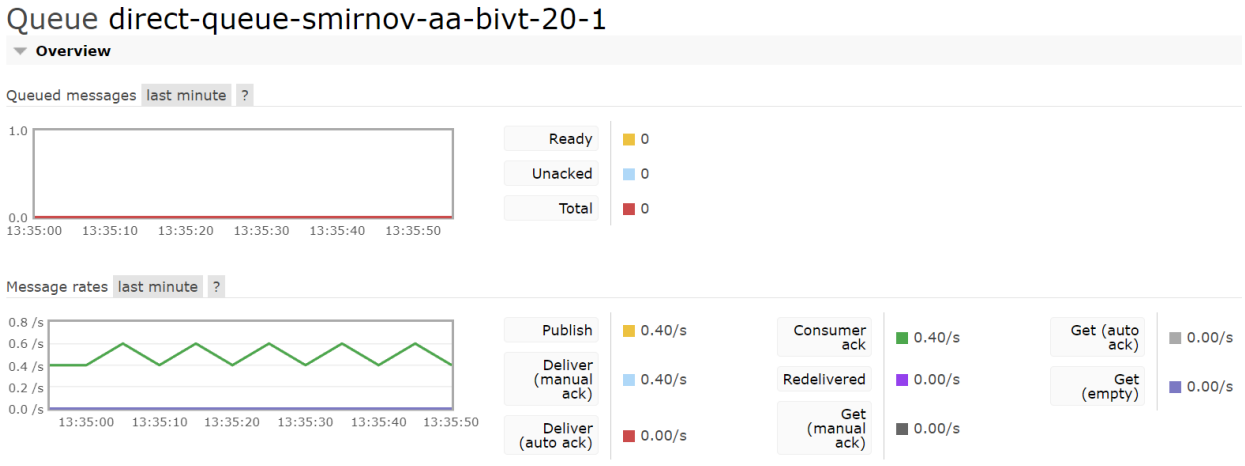

Рисунок 4 – Статистика direct queue.

Вывод: в результате выполнения данной лабораторной работы я познакомился со способом интеграции RabbitMQ с Java Spring. Также выполнил задания, т.е. создал fanout и direct обменники, которые связал с очередями, и протестировал отправку и получение сообщений.