SpringBoot 整合 RabbitMQ，实现消息发送与多个消费者的情况

作者：jockming112
原文链接：https://ld246.com/article/1626784976237
来源网站：链滴

配置

```
### rabbitmq
spring:
  rabbitmq:
    host: 192.168.2.111
    port: 5673
    username: guest
    password: guest
```

依赖

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-amqp</artifactId>
    <version>2.3.3.RELEASE</version>
</dependency>
```

配置队列

```
import org.springframework.amqp.core.Binding;
import org.springframework.amqp.core.BindingBuilder;
import org.springframework.amqp.core.Queue;
import org.springframework.amqp.core.TopicExchange;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

/**
 * 订阅模式
 *
 * @author ming
 * @version 1.0.0
 * @date 2021/2/1 14:49
 **/

@Configuration
public class TopicQueueConfig {

    /**
     * 配置队列
     */
    @Bean(name = "topic_queue")
    public Queue updatePasswordQueue() {
        return new Queue("topic_queue");
    }

    /**
     * 配置交换机
     */
    @Bean(name = "topic_queue_exchange")
    public TopicExchange exchange() {
```

```
        return new TopicExchange("topic_queue_exchange");
    }

    /**
     * 将队列按照相应的规则绑定到交换机上
     *
     * @param queue    消息队列
     * @param exchange 交换机
     */
    @Bean
    public Binding bindingExchangeMessages(@Qualifier("topic_queue") Queue queue,
                                           @Qualifier("topic_queue_exchange") TopicExchange exchange) {
        return BindingBuilder.bind(queue).to(exchange).with("topic.queue.#");
    }
}
```

生产者

```
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.amqp.core.AmqpTemplate;
import org.springframework.stereotype.Component;

/**
 * 生产者
 *
 * @author ming
 * @date 2021/6/22
 **/
@Component
@Slf4j
@RequiredArgsConstructor
public class TopicQueueSender {

    private final AmqpTemplate template;

    /**
     * 发送同步修改密码的队列通知
     *
     * @param content 内容
     */
    public void send(String content) {
        template.convertAndSend("topic_queue_exchange", "topic.queue.update", content);
    }
}
```

消费者

```
import lombok.RequiredArgsConstructor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.amqp.core.ExchangeTypes;
import org.springframework.amqp.rabbit.annotation.Exchange;
import org.springframework.amqp.rabbit.annotation.Queue;
```

```
import org.springframework.amqp.rabbit.annotation.QueueBinding;
import org.springframework.amqp.rabbit.annotation.RabbitListener;
import org.springframework.stereotype.Component;

/**
 * 消费者
 *
 * @author ming
 * @version 1.0.0
 * @date 2021/2/1 11:35
 **/

@Slf4j
@Component
@RequiredArgsConstructor
public class TopicQueueReceiver {

    @RabbitListener(bindings = @QueueBinding(
            value = @Queue(value = "topic_queue_a", durable = "true", autoDelete = "false"),
            exchange = @Exchange(value = "topic_queue_exchange", type = ExchangeTypes.TOP
C),
            key = "topic.queue.update"
    ))
    public void receiverA(String content) {
        log.info("进入监听：{}", content);
    }

    @RabbitListener(bindings = @QueueBinding(
            value = @Queue(value = "topic_queue_b", durable = "true", autoDelete = "false"),
            exchange = @Exchange(value = "topic_queue_exchange", type = ExchangeTypes.TOP
C),
            key = "topic.queue.update"
    ))
    public void receiverB(String content) {
        log.info("进入监听：{}", content);
    }
}
```

注意：

- Exchange和RoutingKey相同、queue不同时，所有消费者都能消费同样的信息；
- Exchange和RoutingKey、queue都相同时，消费者(随机一个)中只有一个能消费信息，其他消费都不能消费该信息。