

# Sprawozdanie laboratorium RabbitMQ

## Zadanie 1a

Potwierdzenie	Restart po przetworzeniu wiadomości	Restart w trakcie przetwarzania wiadomości
Po otrzymaniu wiadomości	TAK	NIE
Po przetworzeniu wiadomości	TAK	TAK

Który sposób potwierdzeń zapewnia większą niezawodność?

Większą niezawodność zapewnia wysyłanie potwierdzenia po przetworzeniu wiadomości, ponieważ w razie restartu wiadomość zostanie zretransmitowana

Co się stanie, jeśli nie będziemy potwierdzać wiadomości ani po otrzymaniu, ani po przetworzeniu?

Wiadomość zostanie wysłana jeszcze raz do konsumenta i znowu zacznie ją przetwarzać.

## Zadanie 1b

Przed

Consumer 1

```
Z1 CONSUMER
Waiting for messages...
Received: 30
End consuming: 30
Received: 1
End consuming: 1
Received: 1
End consuming: 1
Received: 1
End consuming: 1
Received: 1
End consuming: 1
```

## Consumer 2

```
Z1 CONSUMER
Waiting for messages...
Received: 5
End consuming: 5
Received: 5
End consuming: 5
Received: 5
End consuming: 5
Received: 5
End consuming: 5
```

## Po

### Consumer 1

```
Z1 CONSUMER
Waiting for messages...
Received: 1
End consuming: 1
Received: 1
End consuming: 1
Received: 5
End consuming: 5
Received: 1
End consuming: 1
Received: 5
End consuming: 5
```

### Consumer 2

```
Z1 CONSUMER
Waiting for messages...
Received: 5
End consuming: 5
Received: 1
End consuming: 1
Received: 5
End consuming: 5
Received: 1
End consuming: 1
Received: 5
End consuming: 5
```

## Zadanie 2

Direct

Producer

```
Z2 PRODUCER
Enter routing key:
car.orange
Enter message:
test car orange
Sent: test car orange
Enter routing key:
plane.orange
Enter message:
test plane orange
Sent: test plane orange
Enter routing key:
*.orange
Enter message:
test * orange
Sent: test * orange
Enter routing key:
```

Consumer 1

```
Z2_Consumer x Z2_Producer x
/usr/java/jdk-13.0.1/bin/java -javaagent:/usr/local/...
Z2 CONSUMER
*.orange
created queue: amq.gen-67qDsw5huY93CoozCQcvhg
Waiting for messages...
Received: test * orange
|
```

Consumer 2:

```
/slf4j-api-1.7.29.jar:/home/arkadius/studies/sr/
Z2 CONSUMER
car.orange
created queue: amq.gen-HEZ_DYpxGiLJOTUBxu1Zrg
Waiting for messages...
Received: test car orange
```

## Topic

### Producer

```
Z2 PRODUCER
Enter routing key:
car.orange
Enter message:
test car orange
Sent: test car orange
Enter routing key:
plane.orange
Enter message:
test plane orange
Sent: test plane orange
Enter routing key:
*.orange
Enter message:
test * orange
Sent: test * orange
Enter routing key:
```

### Consumer 1

```
Z2 CONSUMER
*.orange
created queue: amq.gen-DFEx4Lv6BXKw9UtG0Z80qQ
Waiting for messages...
Received: test car orange
Received: test plane orange
Received: test * orange
```

### Consumer 2

```
/slf4j-api-1.7.29.jar:/home/arkadius/studies/sr/rat
Z2 CONSUMER
car.orange
created queue: amq.gen--7-0J8fpzP_X-P9gvT0X6g
Waiting for messages...
Received: test car orange
```

Widzimy, że w obu przypadkach zostały użyte te same klucze oraz te same wiadomości. W przypadku routingu Direct wiadomości zostały odebrane tylko jeśli klucz wiadomości dokładnie pasował do klucza zadeklarowanego (znaki specjalne traktowane były standardowo). Każdy z konsumentów otrzymał po jednej wiadomości, gdzie klucz pasował dokładnie. W przypadku routingu Topic znaki specjalne traktowane były zgodnie z opisem i

konsument 1 otrzymał wszystkie wiadomości wysłane przez producenta ponieważ ciąg  
“\*.orange” został dopasowany do wszystkiego.