Search Tutorials

Top RabbitMQ frequently asked interview questions.

In this post we will look at RabbitMQ Interview questions. Examples are provided with explanations.

Q: What is RabbitMQ ?

A:RabbitMQ is an open source message broker software (sometimes called message-oriented middleware) that implements the Advanced Message Queuing Protocol (AMQP). The RabbitMQ server is written in the Erlang programming language and is built on the Open Telecom Platform framework for clustering and failover.

Q: What is an exchange in RabbitMQ?

A: An exchange accepts messages from the producer application and routes them to message queues with help of header attributes, bindings, and routing keys. A binding is a "link" that you set up to bind a queue to an exchange.

Q: What is routing key in RabbitMQ?

A: The routing key is a message attribute. The routing algorithm behind a direct exchange is simple - a message goes to the queues whose binding key exactly matches the routing key of the message.

Q: What is Erlang ? Why is it required for RabbitMQ ?

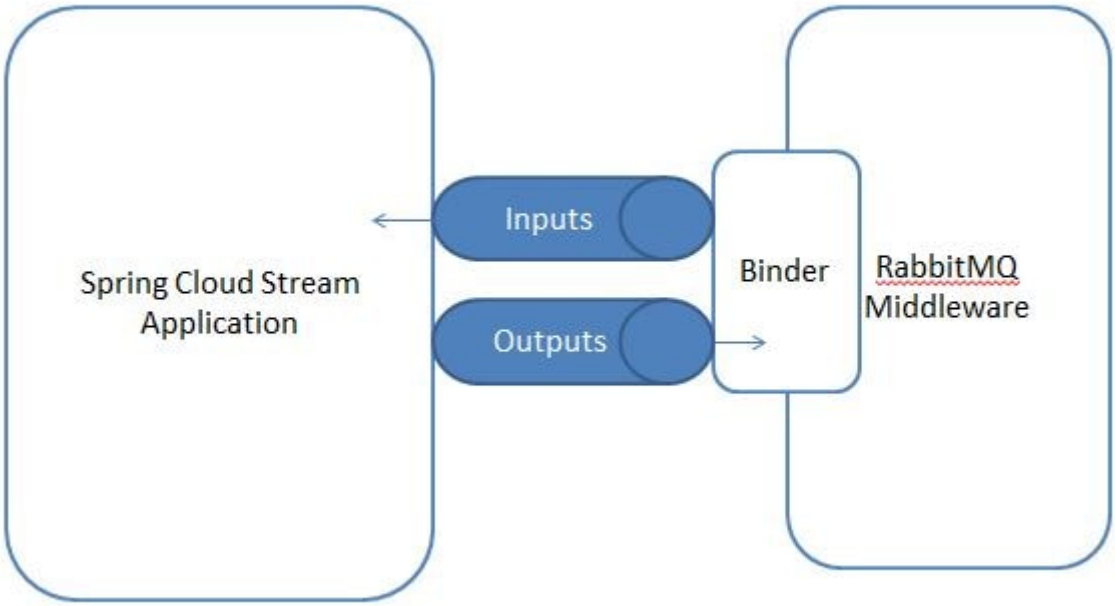
A: Erlang is a general-purpose, concurrent, functional programming language, as well as a garbage-collected runtime system. The RabbitMQ server is written in the Erlang programming language and is built on the Open Telecom Platform framework for clustering and failover. Since RabbitMQ is built on top of Erlang, we will first need to install Erlang beforing installing RabbitMQ **Q: How to install RabbitMQ?**

A:Install RabbitMQ on Windows (/misc/rabbitmq-hello-world)

Q: What is Spring Cloud Stream? What is the need for it?

Q: How to integrate RabbitMQ with Spring Boot?

A: In the previous tutorials we have seen examples for integrating Spring Boot Application with Messaging Systems (like Apache Kafka (/spring/spring-boot-apache-kafka-hello-world) and RabbitMQ (/spring/spring-boot-rabbitmq-hello-world). If you look at these examples these required a lot of configuration code which was Broker specific. For example in case of RabbitMQ integration with Spring Boot (/spring/spring-boot-rabbitmq-hello-world) we had to write code to create AmqpTemplate Template and Bindings. So if tomorrow the Messaging System changes we will also need to make application code changes.
Spring Cloud helps solve this problem using Spring Cloud Stream. Using Spring Cloud Stream we can develop applications where we do not need to specify the implementation details of the messaging system we want to use. We just need to specify the required binding dependencies and Spring Cloud Stream will the integrate the messaging systems to Spring Boot Application.



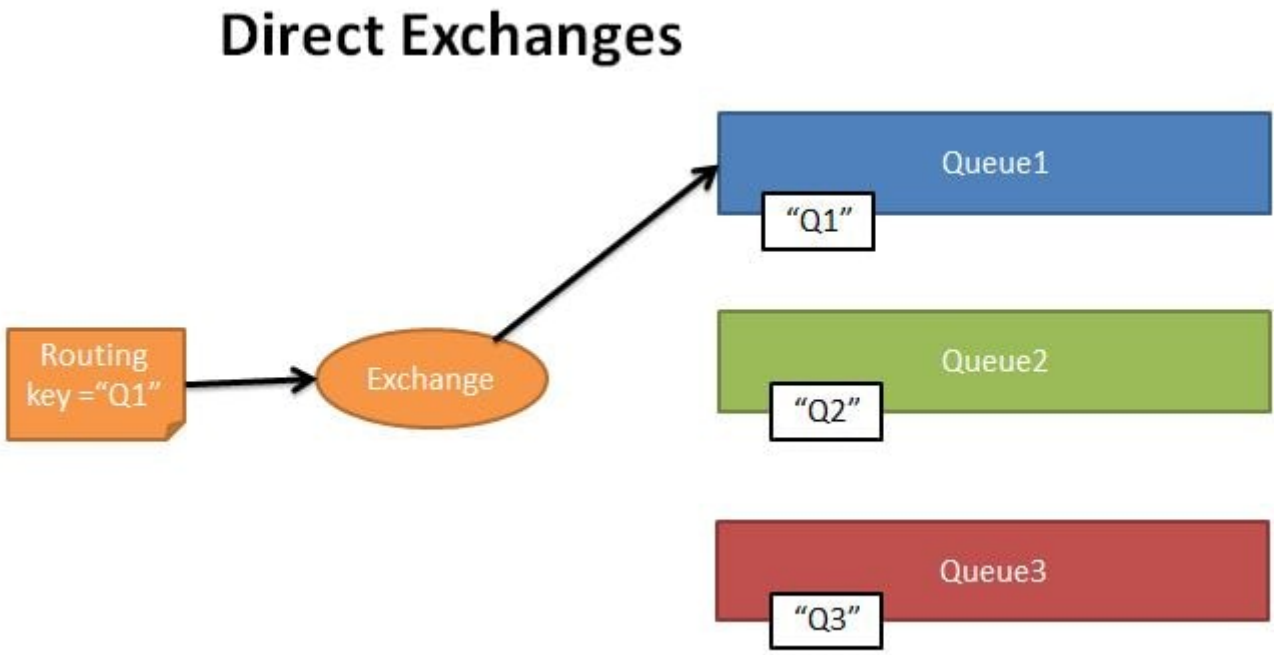
Spring Cloud Stream Tutorial - Publish Message to RabbitMQ Simple Example (/spring/cloud-stream-rabbitmq-1)
Spring Cloud Stream Tutorial - Consume Message from RabbitMQ Simple Example (/spring/cloud-stream-rabbitmq-2)

Q: What are the types of exchanges available in RabbitMQ?

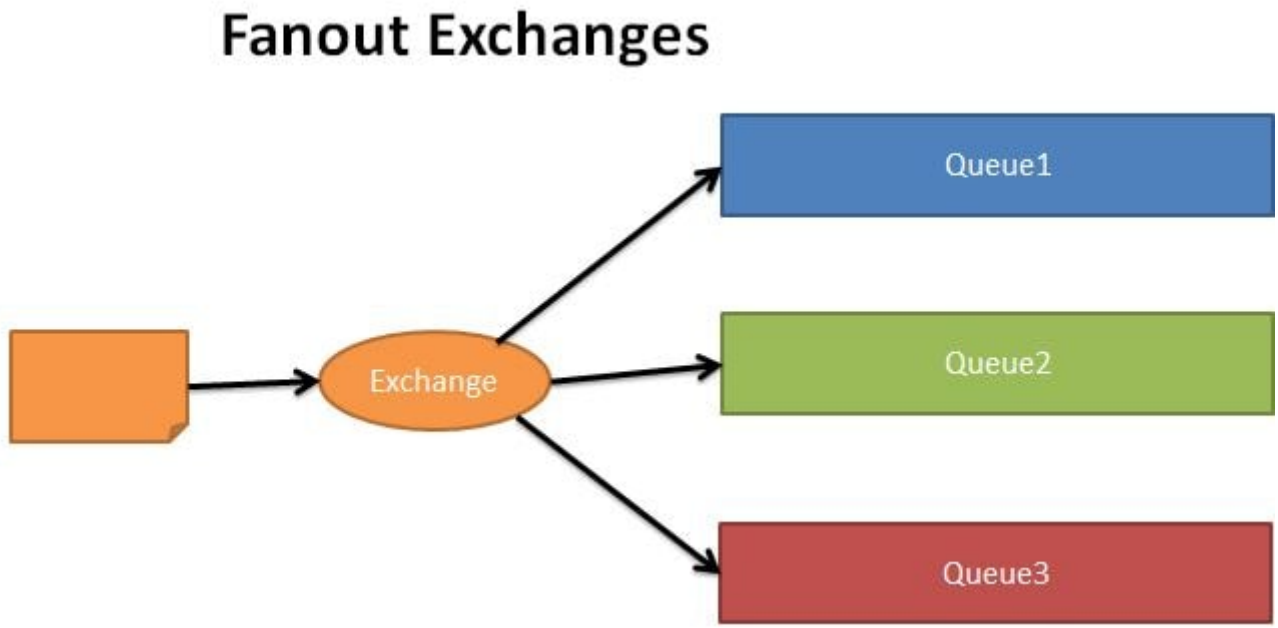
A: There are 4 types of exchanges available in RabbitMQ.

- **Direct -** Based on the routing key a message is sent to the binding queue having the same routing key. The routing key of exchange and the binding queue have to be an

exact match.

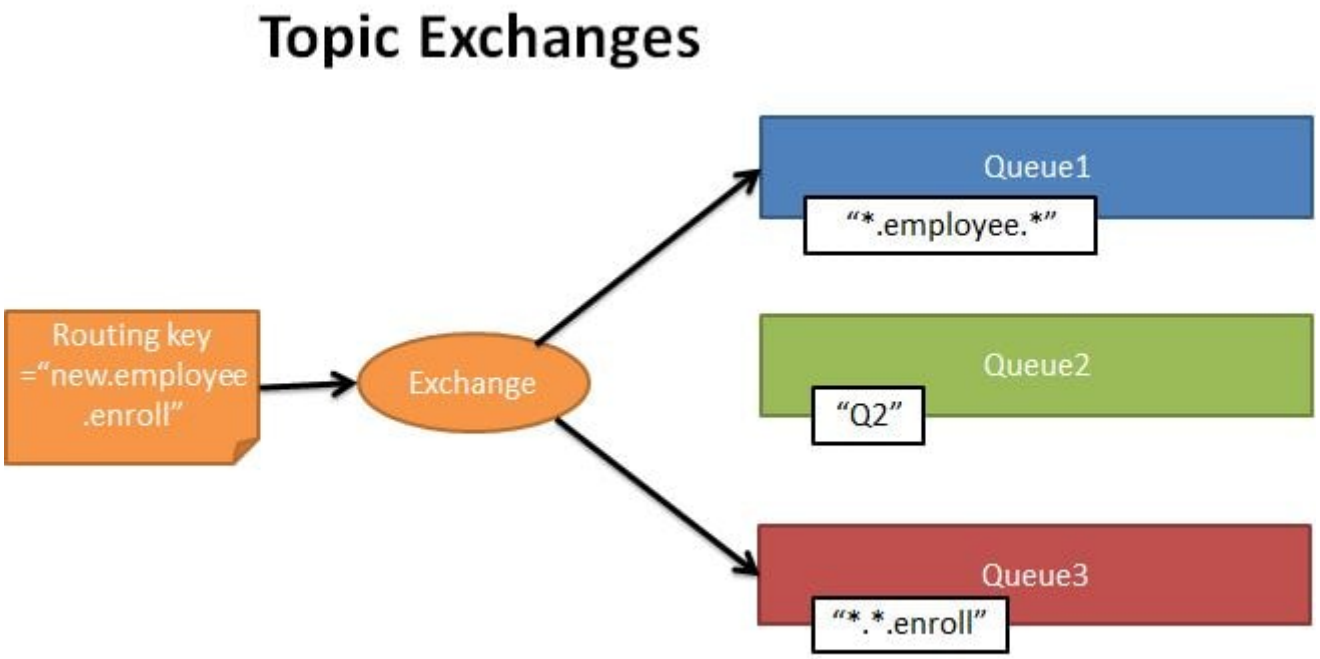


- **Fanout -** The message is routed to all the available bounded queues. The routing key if provided is completely ignored. So this is a kind of publishe-subscribe design pattern.

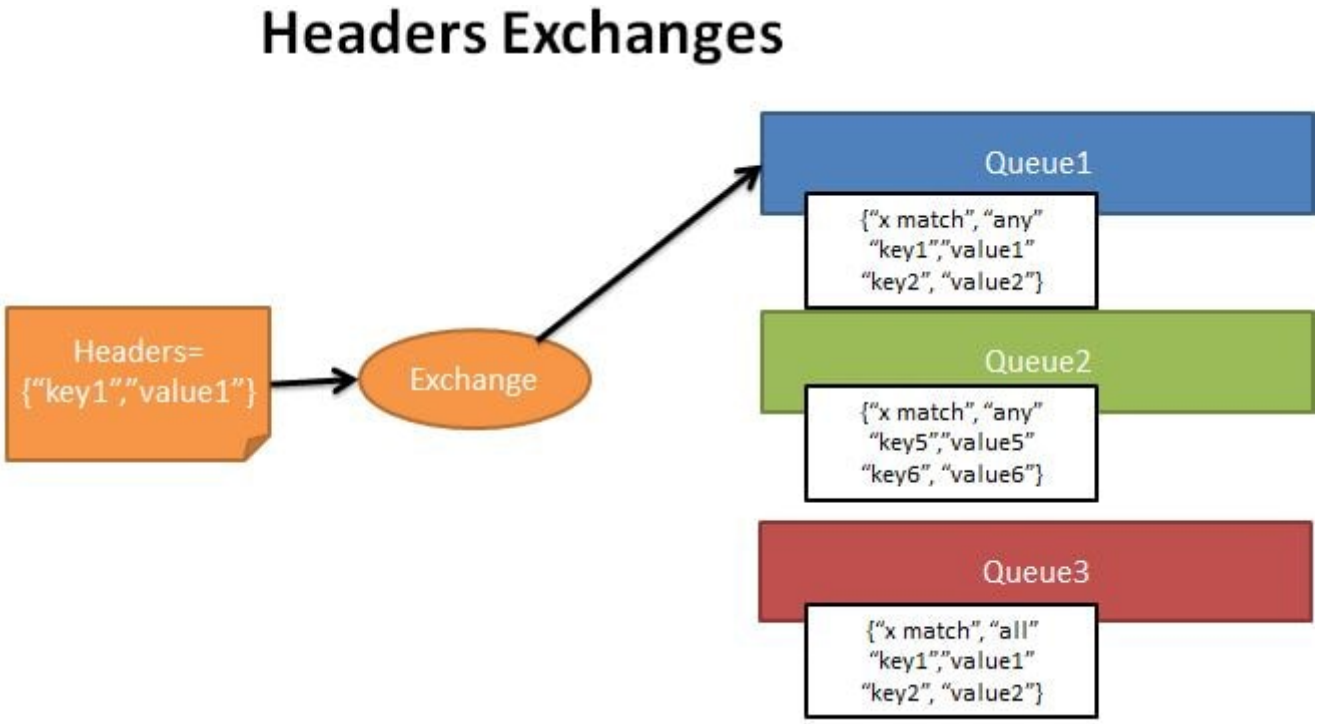


- **Topic -** Here again the routing key is made use of. But unlike in direct exchange type, here the routing key of the exchange and the bound queues should not necessarily be an exact match. Using regular expressions like wildcard we can send the exchange to

multiple bound queues.



- **Headers -** In this type of exchange the routing queue is selected based on the criteria specified in the headers instead of the routing key. This is similar to topic exchange type, but here we can specify complex criteria for selecting routing queues.



Spring Boot + RabbitMQ Tutorial - Implement Exchange Types (/messaging/rabbitmq/exchange)

In this tutorial we will be exploring the various RabbitMQ Exchange types. Also we will be implementing them using Spring Boot.

Q: How to verify RabbitMQ version?

A:

```
sudo rabbitmqctl status
```

Q: How to delete all RabbitMQ queues?

A:

```
rabbitmqctl stop_app
rabbitmqctl reset
rabbitmqctl start_app
```

Q: Does RabbitMQ have any concept of message priority?

A: RabbitMQ does have concept of priority-

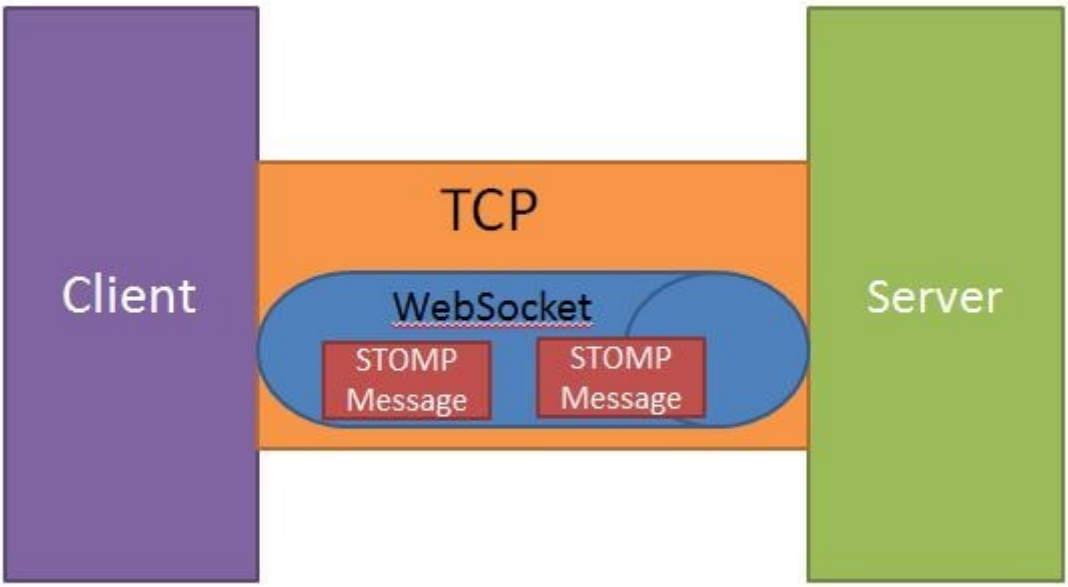
- We can define the queue's priority range at the time the queue is created
- Messages where priority is not set get a priority of 0
- Messages with a numeric priority higher than the maximum set on the queue get the highest priority the queue supports

Q: What is STOMP??

A:STOMP is a simple text-oriented messaging protocol used by our UI Client(browser) to connect to enterprise message brokers.
Clients can use the SEND or SUBSCRIBE commands to send or subscribe for messages along with a "destination" header that describes what the message is about and who should receive it.
It defines a protocol for clients and servers to communicate with messaging semantics. It does not define any implementation details, but rather addresses an easy-to-implement wire protocol for messaging integrations. The protocol is broadly similar to HTTP, and works over TCP using the following commands:

- CONNECT
- SEND
- SUBSCRIBE
- UNSUBSCRIBE
- BEGIN
- COMMIT
- ABORT

- ACK
- NACK
- DISCONNECT



Q: How to implement Chat Application using Spring Boot + WebSocket + RabbitMQ?

A:

Build a Chat Application using Spring Boot + WebSocket + RabbitMQ (/spring/boot-websocket-chat)

Build a web chat using Spring Boot and WebSocket.

Q: How to implement Retry and Error Handling for RabbitMQ?

A:

Spring Boot + RabbitMQ Tutorial - Retry and Error Handling Example (/messaging/rabbitmq/error)

In this tutorial we will be implementing a Spring Boot + RabbitMQ example to retry messages on exception and if exception still exists after maximum retries then put message in a dead letter queue where it can be analyzed and corrected later.