V1.0

Training Program on RabbitMQ Essentials

4P Advisory Services

www.4pa.in

RabbitMQ Essentials Training Program

What is RabbitMQ?

RabbitMQ is an open-source message broker software that is used to transfer messages between applications or services. It was developed by Rabbit Technologies Ltd. and is written in the Erlang programming language. RabbitMQ provides a way for applications to communicate with each other asynchronously, allowing them to work together more efficiently and reliably.

At its core, RabbitMQ implements the Advanced Message Queuing Protocol (AMQP), which is an open standard for messaging middleware. It provides a range of features, including message queuing, routing, and delivery, and supports a variety of messaging patterns, such as publish/subscribe, point-to-point, and request/response. RabbitMQ also provides a web-based management interface that allows administrators to monitor and control message queues and exchanges.

RabbitMQ is widely used in enterprise-level applications and systems, especially in cloud computing environments, where it can help to decouple and scale different components of a distributed system. It is compatible with a variety of programming languages, including Java, .NET, Python, Ruby, and many others, and has a large and active community of users and developers.

Advantages of RabbitMQ as a message broker for distributed systems:

- Asynchronous communication: RabbitMQ supports asynchronous messaging, allowing systems to exchange messages in a decoupled manner. This enables applications to continue processing messages without waiting for a response from the recipient, which can improve overall system performance and reliability.
- Scalability: RabbitMQ can handle large volumes of messages and can be easily scaled by adding more nodes to the cluster. This makes it a good choice for applications that need to handle high message throughput.
- Reliability: RabbitMQ supports message persistence, ensuring that messages are not lost even if a node or network failure occurs. It also supports message acknowledgments, which ensure that messages are delivered successfully before being removed from the queue.
- Flexibility: RabbitMQ supports a wide range of messaging patterns, including pub-sub, point-to-point, and request-response. It also provides a variety of exchange types, such as direct, topic, and fanout, which can be used to route messages between producers and consumers.
- Easy integration: RabbitMQ provides client libraries for many programming languages, making it easy to integrate with existing applications. It also supports a wide range of protocols, including AMQP, STOMP, and MQTT, allowing it to communicate with a variety of systems.

Overall, RabbitMQ provides a reliable and scalable messaging solution that can be easily integrated into a variety of applications. Its support for a wide range of messaging patterns and protocols makes it a flexible choice for distributed systems.

Audience

The RabbitMQ training is a 2-days course designed for:

- Software Professionals
- Developers working in Distributed Systems
- Project Managers and team members executing projects to integrate different systems and protocols

Learning Objectives:

- An understanding of the messaging and queuing concepts that underpin RabbitMQ
- Knowledge of RabbitMQ's architecture and components, including exchanges, queues, and bindings
- Experience with creating and managing RabbitMQ exchanges and queues, and routing messages between them
- Knowledge of advanced messaging patterns such as fanout, direct, and topic exchanges, and the ability to implement them in RabbitMQ
- Understanding of how to handle message acknowledgement and rejection, and how to configure dead letter exchanges.
- Experience with managing RabbitMQ clusters, including users and permissions, monitoring, and troubleshooting.
- Familiarity with RabbitMQ client libraries and how to use them to send and receive messages.
- Knowledge of real-world use cases for RabbitMQ and best practices for deploying and integrating it with other technologies.

By the end of the training program, participants should have a strong understanding of RabbitMQ and be able to deploy and manage RabbitMQ clusters in production environments. They should also be able to use RabbitMQ to implement messaging patterns and integrate them with other technologies.

Candidate Prerequisites

- Basic programming skills: Participants should have some experience with at least one programming language such as Java, Python, or C#.
- Familiarity with Linux command line: Participants should be comfortable working with Linux command-line tools, such as SSH and Bash.
- Networking basics: Participants should have a basic understanding of IP addresses, ports, and network protocols such as TCP and UDP.
- Basic understanding of messaging concepts: Participants should have a basic understanding of messaging concepts such as queuing, routing, and message brokers.
- Knowledge of at least one messaging protocol: Participants should have some knowledge of at least one messaging protocol such as AMQP, STOMP, or MQTT.

While these are the suggested prerequisites, the training program also covers the basics of RabbitMQ and messaging concepts, so participants with less experience may also benefit from the training.

Infrastructure Prerequisites

The hardware and software requirements for practising during the RabbitMQ Essentials training program are as follows:

- A computer with a modern web browser and an internet connection.
- A Red Hat 3scale API Management account: You can sign up for a free trial account on the Red Hat 3scale website. The trial account provides access to all the features of the platform for a limited period.
- A Linux-based virtual machine: Red Hat provides a virtual machine image that you can download and use for the course. The virtual machine comes pre-installed with all the necessary software and tools required for the course, including Red Hat 3scale API Management, OpenShift Container Platform, and other open-source tools.
- OpenShift Container Platform: The virtual machine image comes with a pre-configured OpenShift Container Platform cluster. OpenShift is a Kubernetes-based container platform that enables you to deploy and manage containerized applications and services.
- API development tools: You will need access to API development tools, such as the OpenAPI specification editor, API testing tools, and programming languages and frameworks for API development.
- Cloud infrastructure: To practice deploying APIs in cloud environments, you will need access to a cloud infrastructure, such as Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP).

Overall, the hardware, software, and cloud infrastructure required for practising during the Red Hat 3scale API Management Administration training program are readily available and can be accessed through a web browser with an internet connection. The Red Hat 3scale trial account provides access to all the features of the platform for a limited period, enabling you to practice the course contents in a real-world environment.

Training Outline:

Day 1:

Session 1: Introduction to RabbitMQ

What is RabbitMQ?
Messaging and queuing
RabbitMQ architecture
Activity: Install RabbitMQ and create a basic message queue

Session 2: RabbitMQ Exchanges

Types of exchanges
Binding queues to exchanges
Routing messages with exchanges
Activity: Create fanout, direct, and topic exchanges and route messages to queues

Session 3: RabbitMQ Queues

Durability and persistence
Message properties
Priority queues
Activity: Create durable queues, set message properties, and configure priority queues

Session 4: Advanced Message Routing

Fanout exchanges
Direct exchanges
Topic exchanges
Activity: Use routing keys and binding patterns to route messages between exchanges and queues

Session 5: Message Acknowledgment and Rejection

Message acknowledgment modes
Negative acknowledgments and rejections
Dead letter exchanges
Activity: Implement message acknowledgment and rejection, and configure dead letter exchanges

Day 2:

Session 6: RabbitMQ Administration

Managing users and permissions
Monitoring and metrics
Troubleshooting common issues
Activity: Use the RabbitMQ web interface to manage users and permissions, monitor message queues, and troubleshoot issues

Session 7: RabbitMQ Client Libraries

Overview of client libraries
Using the Java client library
Using the Python client library
Activity: Use client libraries to interact with RabbitMQ and send/receive messages

Session 8: RabbitMQ in Practice

Real-world use cases for RabbitMQ
Best practices for RabbitMQ deployment
Integrating RabbitMQ with other technologies
Activity: Discuss real-world use cases and best practices for RabbitMQ deployment, and explore integrations with other technologies

This two-day program covers all the material in the RabbitMQ Essentials course, with hands-on activities included for each session. By the end of the program, participants should have a good understanding of RabbitMQ and be able to deploy and manage RabbitMQ clusters in production environments