

IBM MQ with Java: Programming and Messaging Foundations

Core administration and a guided Java message-client example

FA-0733

Mainframe & Enterprise Systems · Software Development · DevOps, Cloud & Infrastructure

Introduction

Learn IBM MQ concepts and develop a small Java messaging example with supplied code and configuration. Begin with foundational programming and development tools, then connect client behaviour to queue-manager operations and diagnostic evidence.

The course is introductory and does not imply that every Java framework or production deployment can be covered in depth. Practical work uses a compatible supported JDK, MQ client and isolated local or remote environment; remote access is arranged for the lab rather than promised as a permanent service.

Learning outcomes

- Explain queue-manager, queue and channel roles in point-to-point messaging.
- Inspect and manage selected MQ objects and properties in the supplied lab.
- Trace a distributed messaging example and basic client connectivity.
- Explain triggering, dead-letter handling and cluster concepts.
- Use configuration, reason codes and logs to investigate a selected failure.
- Describe how a client channel definition table supports configured client connections.
- Use basic Java syntax, collections, functions/classes and an IDE in the supplied example.
- Manage a compatible client dependency/build and separate connection configuration from application code.

- Run and inspect a small Java send/receive workflow and explain its failures.

Prerequisites

- Basic computer, file-management and client/server familiarity; no prior Java experience required for the introductory route.
- A supported JDK, suitable IDE and compatible build/client libraries, plus the supplied IBM MQ training environment.
- Approved access to the lab and required network endpoints; SSH is used only for an authorised remote host where the delivery requires it.

Training outline

1. Module 1 — Java and the development environment

- Programming goals, simple algorithms and basic data structures used in the supplied example.
- Java syntax, classes, methods, exceptions and the selected client-code structure.
- Set up the compatible JDK and IDE; inspect Maven/POM dependency management where used.
- Identify the chosen MQ/JMS client API and its supported version; frameworks are options, not interchangeable mandatory dependencies.

2. Module 1 — Messaging and MQ operations

- Point-to-point messaging, queue managers, objects and properties.
- Distinguish client connections, local queues and channels; inspect the supplied environment.
- Use selected MQ command-line tools and verify the resulting object state.

3. Module 2 — Distributed setup and triggering

- Trace a small distributed messaging example and inspect configuration dependencies.
- Introduce triggering and related queues/process definitions in the selected environment.
- Run a supplied message flow and review expected versus observed behaviour.

4. Module 3 — Errors, logs and dead-letter handling

- Interpret selected reason codes and operational logs; distinguish a client problem from a delivery or queue-manager problem.

- Inspect a dead-letter queue and a controlled handling example; review recovery before replaying messages.
- Document evidence, configuration and remaining failure conditions.

5. Module 4 — Clusters and client configuration

- Introduce queue-manager cluster roles and distribution concepts; distinguish clustering from a complete high-availability design.
- Inspect a client channel definition table (CCDT) and how clients locate connection definitions.
- Check client/server compatibility and the authentication/permissions required by the example.
- Review the operational workflow and identify deeper administration or application-development study.

6. Module 6 — Java integration and deployment review

- Configure the Java client, send/receive sample messages and inspect application errors alongside queue-manager logs.
- Keep credentials and endpoint settings separate from the code and apply the required client permissions.
- Run a supplied local or authorised remote deployment example, using SSH where appropriate.
- Review resource cleanup, message-handling assumptions and recovery requirements before any operational use.