

Intermediate Java for Desktop Applications

Collections, concurrency, files and a guided desktop project

FA-0630 · 3 days

Software Development

Strengthen intermediate Java skills through collections, streams, threads, exceptions, file handling and one guided desktop application project.

Introduction

This three-day course is for participants who can already write, build and debug Java programs. It develops data conversion, collection processing, object-oriented design, concurrency and file-handling skills through small exercises.

A guided desktop project brings together selected file and database operations, followed by an assessment. Memory management, desktop interfaces and operating-system integration are covered at introductory project depth; the course does not claim complete JVM expertise or production deployment readiness.

Learning outcomes

The course teaches participants to:

- Explain object/reference lifetime and the role of garbage collection at introductory JVM level.
- Use control flow and bounded recursion for suitable data-conversion tasks.
- Work with collections, generics, lambdas and streams.
- Identify synchronisation and exception-handling issues in a small threaded example.
- Use Java file APIs with appropriate resource and error handling.
- Apply selected database operations in a prepared project.

- Review an introductory Java desktop application, test its behaviour and identify operating-system integration considerations.

Prerequisites

- Ability to write Java programs and use a Java IDE.
- Familiarity with compiling, running, debugging and testing a Java project.
- Basic understanding of the software-development workflow; no claim to know its entire lifecycle is required.
- A prepared compatible JDK, IDE, project and disposable database for the selected exercises.

Training outline

1. Day 1 — OOP and Data Processing

- Revision of OOP
- Loops
- Conditions
- Recursion: base cases, depth limits and situations where iteration is preferable.
- Data conversion, including a bounded nested-data example.
- Primitive and object data handling with explicit conversion checks.
- Validate and clean a small prepared dataset.

Review object/reference lifetime and garbage-collection concepts in the context of the small examples.

2. Day 2 — Collections and Concurrency

- Collections
- Compare lists and arrays, mutability and access patterns.
- Generics
- Lambdas and functional interfaces for small tasks.
- Stream processing and its limitations; do not assume parallel operations improve performance.
- Threads
- Multi-threaded programming
- Synchronisation and shared-state risks in a small example.
- Executor/task utilities appropriate to the selected JDK.

3. Day 3 — Errors, Files and Application Project

- Exception Handling
- Error Handling
- Exception propagation and failure handling in multithreaded examples.
- File and directory operations using `java.nio.file` and related I/O APIs; resource handling and filesystem-specific behaviour.

Guided Project

Extend one prepared application with file processing and selected database operations using sample data; handle resources and errors.

Extend one prepared desktop application with file processing and selected database operations using sample data; handle resources and errors.

Java for the Desktop

Review the selected desktop interface and operating-system-specific application behaviour. Discuss packaging, runtime and dependency considerations for the prepared platform; a full GUI toolkit and multi-platform release process are outside this introductory project.

Review the project's behaviour, explain selected design choices and diagnose a small prepared fault.