Java

Full Stack Trainer Guide

Total Duration of Training

The estimated total training delivery duration of the Java track modules is **29 days**, with each day consisting of **8 hours** of training and 50% of trainer delivery is expected on each day and the remaining time for assignment review and Q/A. However, this estimated duration of the modules is just indicative and can be adjusted as per the size and average learning curve of the cohort as applicable. What is important is not to rush through the course and ensure strong conceptual and practical understanding of each subject.

Guidelines for Trainer –

1) On the first day of the module, share the curriculum outline with the students.
2) Set the expectation that the students must adhere to the module-wise daily plan and complete the daily topics, and assignments on the same day.
3) Emphasize that the implementation of assignments is expected to be completed individually, without assistance from friends or AI tools.
4) Conduct sessions in full ILT mode:
    a) Outline the topics covered each day,
    b) Explain, discuss, and demonstrate complex concepts,
    c) Review the assignments and share feedback, ask the students to present and explain their code
5) Versions expected: JDK 17+, JUnit 5.0, Maven 3.9.9, JPA 3.0, Hibernate 6+, Spring 5.0+, Spring Boot 3+

Teaching Methodology

1) **Interactive Lectures:** Use a mix of lectures and interactive discussions to explain concepts.
2) **Hands-on Practice:** Encourage students to practice coding during the sessions.
3) **Group Activities:** Organize group activities and discussions to enhance understanding.
4) **Quizzes and Assessments:** Conduct regular quizzes and assessments to gauge understanding.
5) **Real-world Examples:** Use real-world examples to explain abstract concepts.
6) **Module-wise assignments** to be shared with the students daily based on the topic covered each day.

Daily Topics Coverage

Java Programming, Debugging, and Unit Testing (10 days)

Day 1:

1. **Development Environment Setup**
    - Installing JDK (Java Development Kit).
    - Setting up IDE (Eclipse/STS).
    - Configuring environment variables (JAVA_HOME, PATH).
2. **Language Fundamentals**
    - Data types (primitive and reference types).
    - Variables (declaration, initialization, scope).
    - Operators (arithmetic, relational, logical, bitwise).
    - Expressions and statements (assignment, compound, control).
3. **Control Flow Statements**
    - Conditional statements (if, if-else, switch).
    - Looping statements (for, while, do-while).
    - Branching statements (break, continue, return).

Day 2 & 3:

4. **Object-Oriented Programming in Java**
    - Classes and objects (definition, instantiation, equality).
    - Inheritance (single, multilevel, hierarchical).
    - Polymorphism (method overloading, method overriding, constructor chaining).
    - Encapsulation (access modifiers, getter/setter methods).
    - Creating and using packages, static imports
    - Abstraction (abstract classes, interfaces).
    - Inner classes (member, local, anonymous).

Day 4:

5. **Exception Handling**
    - Exception hierarchy (checked, unchecked exceptions).
    - Try-catch block (syntax, multiple catch blocks).
    - Throwing exceptions (throw keyword).
    - Custom exceptions (creating user-defined exceptions).
    - Finally block (usage, importance).
    - Try-with-Resources, Single Catch Block for multiple exceptions

Day 5:

6. **Java Collections Framework**
    - Collection interfaces (List, Set, Map, Queue).
    - Implementations (ArrayList, LinkedList, HashSet, TreeSet, HashMap, TreeMap).
    - Iterators (Iterator, ListIterator), Comparable, Comparator.
    - Utility classes (Collections, Arrays).
    - Generics (generic classes, methods, bounded types).

Day 6:

7. **Java New Features**
    - Lambda expressions (syntax, functional interfaces).
    - Streams API (creating streams, intermediate and terminal operations).
    - New Date/Time API (LocalDate, LocalTime, LocalDateTime, DateTimeFormatter).
    - Java.util.Optional (for wrapping of return values)

Day 7:

8. **Input/Output in Java**
    - File handling (File class, creating/deleting files).
    - Streams (InputStream, OutputStream, Reader, Writer).
    - Buffered streams (BufferedReader, BufferedWriter).
    - Serialization (Serializable interface, ObjectInputStream, ObjectOutputStream).

- Introduction to Java NIO package and asynchronous input (Not for exam)

Day 8:

9. **Java Memory Management**
    - Garbage collection (automatic memory management).
    - Memory leaks (causes, prevention).
    - JVM memory model (heap, stack, method area).
10. **Concurrency in Java**
    - Threads (creating, starting, joining threads).
    - Synchronization (synchronized methods, blocks).
    - Concurrency utilities (ExecutorService, Callable, Future, CountDownLatch).

Day 9:

11. **Debugging and Unit Testing**
    - Debugging techniques (breakpoints, watchpoints, step execution).
    - Unit testing with JUnit (annotations, assertions, test cases).

Day 10:

12. **Database Manipulation with JDBC**
    - JDBC API (DriverManager, Connection, Statement, ResultSet).
    - Executing SQL queries (select, insert, update, delete).
    - Handling result sets (iterating, processing data).
    - Prepared statements (parameterized queries, batch updates).
13. **Logging in Java**
    - Importance of Logging
    - Logging vs Debugging
    - Introduction to Log4j
    - Log levels and Categories

Apache Maven (1 day)

1. **Maven Fundamentals**
    - Introduction to Maven (build automation tool).
    - Benefits of using Maven (dependency management, project structure).
2. **Software Setup**
    - Installing Maven (command line, IDE integration).
    - Configuring Maven (settings.xml, environment variables).
3. **pom.xml and Directory Structure**
    - Understanding pom.xml (project object model).
    - Standard directory layout (src/main/java, src/test/java).
4. **Multi-Module Project Creation**
    - Creating multi-module projects (parent POM, child modules).
    - Managing dependencies across modules.
5. **Scopes and Dependency Management**
    - Dependency scopes (compile, provided, runtime, test, system).
    - Managing transitive dependencies (exclusions, dependency management).

JPA with Hibernate (4 days)

Day 1:

1. **Introduction to JPA & Hibernate**
    - Introduction to Hibernate
    - Basics of JPA (Java Persistence API).
    - Setting up JPA with Hibernate (configuration, annotations).
2. **Implementing CRUD using JPA**
    - Creating entities (annotations, mapping).
    - Reading entities (JPQL, criteria API).
    - Updating entities (merging, detaching).
    - Deleting entities (removing, cascading).

Day 2:

3. **Establishing Relationships with JPA**
    - One-to-one relationships (mapping, annotations).
    - One-to-many relationships (mapping, annotations).
    - Many-to-many relationships (mapping, annotations).

Day 3:

4. **Queries with Entities using JPQL**
    - Writing JPQL queries (syntax, keywords).
    - Named queries (annotations, usage).
    - Criteria API (dynamic queries, type-safe queries).

Day 4:

5. **Caching with Hibernate & JPA**
    - First-level cache (session cache).
    - Second-level cache (configuration, providers).
6. **Transaction Management**
    - Managing transactions (begin, commit, rollback).

    (Note: All coding demos/assignments will be using JPA APIs and import packages and no Hibernate-specific classes/interfaces will be used. Hibernate will just be used as an implementation)

<u>Spring Framework (3 days)</u>

Day 1:

1. **Introduction to Spring Framework**
    - Overview of Spring (core concepts, benefits).
    - Spring modules (core, AOP, MVC, data access).
2. **Layered Architecture of Spring Framework**
    - Understanding layered architecture (presentation, business, data access layers).
3. **IOC & DI, Configuration of IOC**

- Inversion of Control (IoC) (concept, benefits).
- Dependency Injection (DI) (setter, constructor injection).
- Configuration methods (XML, annotations, Java code).

Day 2:

4. **Various Approaches for DI**
    - Setter injection (usage, examples).
    - Constructor injection (usage, examples).
    - Auto-wiring (types, configuration).
5. **Bean Scopes & Life Cycle**
    - Bean scopes (singleton, prototype, request, session, global session).
    - Bean life cycle (initialization, destruction, lifecycle callbacks).

Day 3:

6. **Spring AOP**
    - Aspect-Oriented Programming (AOP) (concepts, benefits).

Spring Boot (6 days)

Day 1:

1. **Introduction to Spring Boot**
    - Overview of Spring Boot (features, benefits).
    - Setting up Spring Boot (initializers)
    - Spring Boot – Configuration
    - Creating console applications (Command Line Runner, Application Runner).

2. **Spring MVC Fundamentals**
    - Introduction to Spring MVC
    - Controllers, Models, and Views
    - Request mappings and handling
    - Using Thymeleaf templates

Day 2, 3 and 4

1. **Spring Boot REST API**
    - Building REST APIs (controllers, request mapping, response handling).
    - Integrating Spring Data JPA (repositories, CRUD operations).
    - JPA Transactions (ACID properties, propagation, isolation)
    - RestTemplate
    - Spring Boot Global Exception Handling(Handling exceptions globally (ControllerAdvice, ExceptionHandler).

Day 5:

2. **Spring Security**
    - Introduction to Spring Security:
    - Setting up Spring Security in a Spring Boot project
    - Authentication:
        1. Basic authentication
        2. Form-based authentication
        3. Introduction to OAuth2 and OpenID Connect
    - Authorization:
        1. Role-based access control (RBAC)
        2. Method-level security

Day 6:

3. **Spring Unit Testing**
    - Unit testing with JUnit and Mockito (annotations, test cases, mocking).

<u>Version Control using GIT (2 days)</u>

Day 1:

1. **Key Concepts of Git**

- Basics of Git (version control, repositories).
- Git workflow (staging, committing, pushing, pulling).
- Git / Bitbucket Integration with Eclipse / STS

2. **Git Workflow**
    - Understanding the Git workflow (local, remote repositories).
3. **Comparing States in Git**
    - Comparing branches (diff, merge).
    - Comparing commits (log, diff).

Day 2:

4. **Managing Files with Git**
    - Moving, renaming, deleting files (commands, best practices).
5. **Updating Files Managed Outside Git**
    - Handling external file updates (tracking, ignoring).
6. **Creating and Forking Repositories**
    - Using GitHub or Bitbucket (creating repositories, cloning, forking).
7. **Creating Branches and Resolving Merge Conflicts**
    - Branch management (creating, switching, merging).
    - Conflict resolution (identifying, resolving).

Introduction to DevOps and CI/CD Pipeline (3 days)

Day 1:

1. **Overview of Continuous Integration & Continuous Delivery**
    - CI/CD concepts (benefits, practices).
    - Tools and technologies (Jenkins, Bitbucket Pipelines).
2. **Introduction to Jenkins for CI**
    - Setting up Jenkins (installation, configuration).
    - Creating and managing Jenkins jobs (freestyle, pipeline).
3. **Implementing CI Pipeline**

- Using Jenkins or Bitbucket Pipelines (configuration, scripting).

Day 2:

4. **Introduction to Containerization and Docker**
    - Docker basics (concepts, architecture).
    - Docker tools (Docker CLI, Docker Compose).
5. **Creating and Managing Docker Images**
    - Building Docker images (Dockerfile, best practices).
    - Managing Docker images (tagging, pushing, pulling).
6. **Managing Docker Containers**
    - Running, stopping, removing containers.
    - Inspecting and logging containers.
    - Managing container volumes and networks.

Day 3:

7. **Introduction to Docker Networking**
    - Networking concepts in Docker (bridge, host, overlay networks).
    - Configuring and managing Docker networks.
8. **Kubernetes Overview**
    - Basics of Kubernetes (architecture, components).
    - Deploying and managing applications on Kubernetes.
    - Kubernetes services, pods, deployments, and namespaces.

Nature of the Exam

Sections	Module	MCQ	Coding	Marks	Time
Section 1	Java Programming, Unit Testing and Debugging	10	1 Coding Development	20	180 Minutes (3 hrs)
Section 2	JPA with Hibernate + Spring Framework	20	N/A	20	
Section 3	Spring Boot Web Services	10	1 Coding Full Stack	20	
Section 4	Version control using GIT + Introduction to DevOps and CI/CD Pipeline	10	N/A	10	

Thank You