TeamLease Digital

Course Outlines

Java Track

Java Track Training

JAVA Version 11

Day-1

- Introduction to Java
- Introduction of JDK, JRE, JVM
- Data Types, Variables, and Arrays
- IDE overview – Eclipse.
- String & String Methods
- Operators
- Control Statements – if, if-else, Switch, For, While, Do While, Break, Continue, java Comments.
- OOPs - Object, Class, Java keywords, Attributes, Methods, Inheritance (Is-A & HAS-A), Polymorphism, Abstraction, Encapsulation, Constructor, Static, this keyword.
- Access Modifiers – Private, Public, Protected
- Overloading & overriding.
- Wrapper Classes.
- Inner & Nested Classes.
- Object Cloning.

Day-2

- Packages and Interfaces
- Exception Handling - Try-catch, Multiple Catch Block, Nested try, Finally, Throw Keyword, Exception Propagation, Throws Keyword, Throw vs Throws, Final vs Finally vs Finalize Exception, Custom Exception.

Day-3

- Multithreading - Life Cycle of a Thread, Create Thread, Thread Scheduler, Sleeping a thread, Start a thread, Calling run() method, Naming a thread, Thread Priority, Daemon Thread, Thread Pool, Thread Group Shutdown, Garbage Collection. Executor Framework basics.
- Features of Java 5 - For-each loop, Varargs, Static Import, Autoboxing and Unboxing, Enum, Covariant Return Type, Annotation, Generics.

Day-4

- Features of Java 8 - Lambda expressions, Method references, Functional interfaces, Stream API, Default methods, Base64 Encode Decode, Static methods in interface, Optional class, Collectors class, ForEach() method, Parallel Array Sorting, IO Enhancements, Concurrency Enhancements, JDBC Enhancements.

Day-5

- Java Memory Model (Memory Management) and GC
- Java IO & Reflection.
- Java Collections – Map, Set, List, Enumerators, Comparator & Comparable

Day-6

- Collections Stream API's.
- Overview of Concurrency API

JDBC Overview

- Types of JDBC Drivers
- JDBC Classes/Interfaces & Connectivity Steps + CRUD Operations

Kafka: 3 Days

Day-7

a) Kafka Fundamentals

- Install and Configure Kafka
- Kafka Basics
- Kafka Message Flow
- Kafka Basic Operations
- Kafka Architecture
- Kafka Architecture Overview
- Core Kafka Components
- Role of Zookeeper
- Topics, Partitions & Offsets
- Brokers & Topics
- Topic Replication
- Kafka Producer & Consumer
- Producers & Message Keys
- Consumers & Consumer Groups
- Consumer Offsets & Delivery Semantics

- Kafka Broker Discovery
- Consumer Group
- Kafka Design
- Design Strategies
- Kafka Physical Storage
- Partitioning Strategies
- Kafka Replication and Leader Election
- Managing Consumer Offset
- Kafka Cluster using Docker Containers

b) Kafka Extended APIs

- Kafka Connect
- Kafka Streams
- Kafka Schema Registry
- **KSQL**

Day-8

c) Kafka Connect

- What is Kafka Connect?
- Kafka Connect Architecture
- Connectors, Configuration, Tasks, Workers
- Standalone & Distributed Mode
- Kafka Connect Source Connectors (FileStream,JDBC & others)
- Kafka Connect Sink Connectors
- Kafka Connect UI
- Kafka Connect REST API
- Writing our own Kafka Connector

d) Kafka Streams

- Kafka Streams Introduction
- What is Kafka Streams?
- Stream Processing Topology
- Creating source streams from Kafka
- Transform a stream
- KStream & KTable : Stateless transformations
- KStream & KTable : Stateful transformations
- KStream & KTable Transformations (MapValues/Map, Filter/Filter Not,FlatMapValues/FlatMap,Branch,SelectKey
- Aggregating

- Joining
- Join co-partitioning requirements
- KStream-KStream Join
- KTable-KTable Join
- KStream-KTable Join
- KStream-GlobalKTable Join
- Join Constraints - Co-partitioning of Data
- Windowing
- Tumbling time windows
- Hopping time windows
- Sliding time windows
- Session Windows
- Applying processors and transformers (Processor API integration)
- Reading from Kafka (KStream & KTable)
- Writing streams back to Kafka (KStream Vs KTable)
- KStream & KTable Duality
- KTable as Stream(changelog) & KStream to reconstruct KTable
- Transforming KTable to a KStream
- Starting a Kafka Streams application (Running Streaming Application)
- Elastic scaling of your application
- Kafka Streams Vs Spark Streaming
- When to use KStream Vs KTable

Day-9

d) Kafka Schema Registry

- Kafka Confluent Schema Registry
- Need for a Schema Registry
- Confluent REST Proxy
- Kafka AVRO Record Schema
- Compatibility Rules
    - FORWARD Compatibility
    - Backward Compatibility
    - Full Compatibility
- Confluent Schema Registry Operations

e) KSQL

e) KSQL - Data Processing Pipelines in Kafka

- KSQL Setup & Introduction
- How does KSQL work?

- KSQL Command Line
- KSQL Stream
- KSQL Table
- Push & Pull Queries
- KSQL Joins
- Kafka Connect with ksqlDB
- **Windows in KSQL (Tumbling,Hopping,Session)**

Unit Testing Tools – Junit & Mockito

Day-10

Junit

- Junit 5 Architecture
- Environment setup
- Working with JUnit 5
- Creating Testcases
- JUnit 5 Annotations
- JUnit Assertions
- Assumptions
- Suite Tests
- Using @Test in Junit5
- Using Annotations - @BeforeAll and @AfterAll
- TestFixtures with @BeforeEach and @AfterEach
- Testing Exceptions using assertThrows
- Combining Test Cases as TestSuite
- Using @RepeatedTest
- Tagging and filtering

Day-11

Mockito Framework

- Mockito – Overview & Environment Setup
- Mockito - JUnit Integration
- Adding & Verifying Behavior
- Expecting & Varying Calls
- Mockito - Exception Handling
- Mockito - Create Mock & Ordered Verification
- Mockito - Callbacks
- Mockito - Resetting Mock
- Behavior Driven Development
- Mockito - Timeouts

Day-12

JSP

- JSP Fundamentals – Expressions, Scriptlets, Declarations
- JSP Built-In Objects

- Reading HTML Form Data with JSP
- State Management with JSP

Struts

- Evolution of web applications and pitfalls with model-1 architecture
- Motivation for struts framework
- MVC design pattern
- Introduction to Struts framework

Spring

- Spring – Overview & Architecture
- Environment Setup
- Spring Modules.
- Spring - IoC Containers
- Spring - Bean Definition, Scopes, and Life Cycle
- Spring - Bean Post Processors
- Spring - Bean Definition Inheritance
- Spring - Dependency Injection
- Spring - Injecting Inner Beans
- Spring - Injecting Collection
- Spring - Beans Auto-Wiring
- Annotation Based Configuration
- Spring - Java Based Configuration
- Spring - Event Handling & Custom Events in Spring
- Spring - AOP with Spring Framework
- Spring - JDBC Framework
- Spring - Transaction Management
- Spring - Logging with Log4J
- OAUTH vs Auth2
- Spring Security

Day-13

Spring Boot

- Spring Boot Architecture.
- Spring Vs Spring MVC Vs Spring Boot.
- Spring Initializr.
- STS.
- Spring Project Components – SB Annotations, Dependency management, Application Properties, Spring boot Starter, SB Starter Parent, SB starter Web, SB packaging, SB Auto-Configuration.

- Tool Suite – project deployment.
- Spring Boot – AOP.
- Caching – Spring Boot EHCaching.
- Spring Boot Hello World Example using Gradle.

Day-14

Spring RESTful web service

- Spring REST Hello World Example
- Spring REST Validation Example
- Spring REST + Spring Security Example
- Spring REST Error Handling Example
- Spring REST Integration Test Example

Externalized Configuration

- Spring Boot @ConfigurationProperties example
- Spring Boot YAML examples
- Spring Boot Profiles example
- Spring Boot Profile based properties and YAML example

Logging

- Spring Boot Log4j 2 example
- Spring Boot SLF4j Logback example

Database Connectivity

Day-15

DBC Revisit

- Different types of drivers connect with postgres
- Issues with JDBC / Typical JDBC Flow
- Introduction to ORM & ORM Frameworks

Hibernate and JPA

- Hibernate Basics
- Features of Hibernate
- Installation and Environment Setup
- Hibernate Architecture Overview
- Configuration and Session Factory
- Session & Transaction
- CRUD Examples
- Spring Data JPA examples.

- Spring Boot + Spring data JPA
- Spring Boot + Spring data JPA + MySQL
- Spring JDBC JdbcTemplate examples.

Day-16

- Spring Boot JDBC Examples
- Spring Boot JDBC Stored Procedure Examples
- Spring JdbcTemplate Querying Examples
- Spring JdbcTemplate Handle Large ResultSet
- Spring JdbcTemplate batchUpdate() Example
- Spring Boot JDBC Image BLOB Examples
- Spring Boot + Spring Data JPA + Oracle example

Spring Microservices

Day-17

- Introduction to Microservices
- What is Microservices?
- Why Microservices?
- Characteristics of Microservices
- Patterns in Microservices Architecture
- Spring security concepts
- Authentication and Authorization
- Oauth2 concepts (just overview)

Spring MVC Framework

- Introduction to Spring Web MVC
- Spring MVC Architecture
- MVC Architecture
- Dispatcher Servlet
- Creating a spring application using Spring Boot (Already Covered)
- Use of Controllers, View Resolvers
- Stereotypes: @Component, @Service, @Controller, @Repository
- Consume Spring REST Service from Spring MVC
- Spring Data JPA concepts
- Spring MVC Hello World using Maven.

Day-18

Day-19

Web Services SOAP & REST

- Introduction to web services
- What are web services?
- SOAP and Rest Overview.
- Service-Oriented Architecture
- Architecture of web services
- REST Architecture & Maturity Model
- HTTP Methods, Status codes, HTTP Headers, Idempotency, HTTP2
- Producing and consuming a restful web service
- Designing REST API
- REST API Error Handling Patterns
- REST API Response Data Handling Patterns
- REST API Security
- REST API Specifications using Swagger 2.0 / OAI
- REST API Architectural Constraints

Day-20

Swagger Tools

- Overview of Swagger Tools
- Swagger vs OpenAPI
- SwaggerHub - An Overview
- Swagger Editor & Swagger UI - Creating New API
- Swagger Codegen - Generating Server Stub
- Swagger Inspector: Testing API Server
- Auto-generating the Swagger file from code annotations
- Integrating REST with Swagger
- Creating REST API from Swagger Spec
- Generate REST client with Swagger Spec

Day-21

Open API

- OpenAPI Introduction
- Overview of OpenAPI
- Defining a Microservice with OpenAPI
- OpenAPI Schema
- OpenAPI Components
- OpenAPI Parameters
- OpenAPI Requests
- OpenAPI Security Definitions

- OpenAPI Code Gen

Day-22

Okta

- Okta Overview
- Defining Users in Okta
- Configuring External Directories
- Groups in Okta
- Configuring SSO & Provisioning
- Configuring Custom App Integrations
- Okta Policy Framework

Day-23

DevOps Deep-Dive: 5 Days

Introduction to Bitbucket

- What is Bitbucket
- Bitbucket Features
- Dashboard
- Forking
- Command line guidance
- Pull requests

Basic Git Operations

- Using Git
- Commit
- Viewing History
- Configuring Git
- User Identification
- .gitignore

Branching, Merging and Remotes

- Branches in Git
- Branches in Git (cont'd)
- Merge
- Fast Forward Merge
- JUnit with Annotations
- JUnit Design

- Testing Strategies
- Testing Simple Working with Remotes
- Fetch and Pull
- Push
- Pull Requests
- Tagging a Commit

Introduction To GitFlow

- What is GitFlow
- Benefits
- How GitFlow works?
- GitFlow Extension
- Initializing GitFlow
- Features
- Release
- Hotfixes

Day-24

Getting Started with Maven

- Terminology and Basic Concepts
- Artifacts
- Lifecycle
- Plugins
- Running Maven from an IDE
- Common Goals
- pom.xml
- Standard Layout for Sources

Introduction to Gradle

- What is Gradle
- Why Groovy
- Build Script
- Sample Build Script
- Task dependency
- Plugins
- Dependency management
- Gradle Command-line Arguments

Day-25

Jenkins Installation and configuration

- Installation types and product architecture
- Jobs in Jenkins
- Parameterized Jobs
- Built in Environment Variables
- Configuration matrix /Role Based Access

Monitoring Jobs

- Build Pipeline
- Create Build pipeline
- Build file (Scripted pipeline)

Job Types in Jenkins

- Different types of Jenkins Items
- Configuring Source Code Management (SCM)
- Working with Git
- Build Triggers
- Schedule Build Jobs
- Polling the SCM
- Polling vs Triggers
- Maven Build Steps

Test driven development (TDD)

- Test, code, refactor, repeat
- The ROI of TDD
- Rationale
- Advantages
- Tools

Continuous Code Quality using SonarQube

- Understanding SonarQube
- SonarQube - Benefits
- Seven Axes of Quality
- Potential Bugs
- Tests
- Comments and Duplication
- Architecture and Design
- Complexity
- SonarQube Components
- Code Quality (LOC, Code Smells)

- Code Quality (Project Files)
- Code Quality (Code)

Artifact Management

- Repository Manager
- Proxy Remote Repositories
- Types of Artifacts
- Release Artifacts
- Snapshot Artifacts
- Reasons to Use a Repository Manager
- Repository Coordinates
- Publishing Artifacts using Maven

Artifact Management - Security using Xray

- Introduction to DevSecOps
- What is Xray?
- Security & Compliance with Xray
- Component Graph with Xray
- Xray Policies
- Xray Watchers

Day-26

Docker

- Introduction to Docker
- Linux croups and namespaces
- Introduction to Containers
- introduction to Docker
- Docker EcoSystem
- Docker Components
- Docker Architecture
- Docker Engine Components and Dataflow
- Classroom Environment With Docker setup
- Containers Creations
- Containers Operations
- Docker – Images Operations
- DockerHub
- Registry and Repositories in Docker
- Docker - Building Images
- Image Build with Dockerfile
- Deep Dive (Advanced) – Images

- Real time application image builds
- Deep Dive – Containers
- Housekeeping (Deleting dangling images, containers and volumes)
- Container Network Model
- Docker Volumes
- Persistent data in Docker Volumes
- Docker Compose

Day-27

Kubernetes

- Why Kubernetes
- Kubernetes Cluster objects overview
- Kubernetes Cluster Installation Methods
- Kubernetes communication and Architecture
- Kubernetes Various components: - Apiserver
- Etcd
- Controller Manager
- Kube Scheduler
- Kube Proxy
- Kubelet
- Kubernetes Pods various implementation operations
- Kubernetes Labels and Selectors various implementation operation
- Kubernetes Replicasets various implementation operations
- Kubernetes Deployments various implementation operations
- Kubernetes Services various implementation operations
- NodePort
- Cluster IP
- Load Balancers
- Kubernetes object deletion process
- Secret
- Resource Quota

SAST tool (Snyk and CheckMarx)

React: 4 Days

Day 28

Introduction to React

What problem(s) does React solve?

- Traditional, pre-JS web applications
- Late-model, MV* and JS web applications

React's solutions

- Single-page apps
- View libraries
- Helper libraries

React development environment

- Simplicity:create-react-app
- Build-your-own: an overview

Hello world

- Your first React component
- Using React within a page
- Making some basic changes
- React and JSX

JSX

- What is JSX?
- Using JSX
- Using React with JSX
- Using React without JSX
- Precompiled JSX

Day 29

Components

Two types of components

- Functional components
- Class-based components
- Why use one or the other?

Props and state

- Passing in properties
- Limitations of properties
- Using state
- When to use state, when to use props

Event handling

- React event handling

- Synthetic events
- Reactvs DOM event handling

Children

- Components within components
- Known children and unknown children
- Testing child components

Parent-child component communication

- Communication from parent to child
- Communication from child to parent
- Container presentational components

Day 30

React Component Lifecycle

- Overview
- Startup and mounting
- Updating
- Unmounting
- Calling lifecycle methods in tests
- Error handling and error boundaries

Intermediate component usage

- PropTypes
- Typing and React
- Using PropTypes
- PropTypes in production

Asynchronous data

- When should asynchronous fetching be done?
- What challenges does async offer?
- Asynchronous best practices

Lists of data

- Iterating over a list
- The key property
- Sorting data

Day 31

Forms

- Controlled vs uncontrolled components
- What does React now about your form field?
- Does React control your form field?
- When does React find out about changes to your form field? Form field types

Controlling a text field

Other form fields

Getting data out of a form

Working with form data in tests

- Introduction to routing
- What problem is routing trying to solve?
- How does routing solve this problem?
- Tying components to URLs
- Passing parameters via the URL

Routing software

- React-router
- Other routers
- Simple router example
- More complex routing o Top-level routing
- Routing at the top of your application
- Allowing other parts of the application to manage routing

Redirects

React-router objects

- match
- history
- location
- Routing organizational techniques

Hooks

- What are hooks, and what problem do they solve?
- Defined hooks
- The state hook
- The effect hook
- Rules of hooks
- Using hooks today

- Advanced React

Understanding and optimizing reconciliation

- Best practices for React reconciliation
- Recognizing common issues
- Making improvements

React.js Best Practices

- Refs
- What's a ref?
- What problem does it solve?
- How can I use refs?
- The challenges of testing refs Context
- What is the context API?
- Is the context API public?
- How to use the context API

Add-ons –

- Design pattern in React
- Migration from old react version to latest version + Local setup
- Micro Front end architecture using webpack –
- Overview of State management.