Java Training for New Hires

Duration: 15 days
@Vinod – Can this be done from 9am to 3pm ?

Pre-requisites for attending this training:

- Basic knowledge of any programming language (like C or C++)
- Good understanding of how to use SQL commands on RDBMS tables
- Basic knowledge of HTML/CSS/JavaScript
- During the training, the students will be given online resources for self learning, and students are expected to learn few topics by making use of the given resources.

Software setup required in participant's PC/Laptop:

- Java development kit Version 8 (JDK 1.8)
- MySQL (preferred) Version 8
- Eclipse for Java EE developers
- Apache Tomcat 9 (ZIP file)
- Open internet connection is required for downloading project dependencies during training via Maven

Week wise breakup:

- Week 1: Core Java
- Week 2: Adv. Java
- Week 3: Java EE (Web applications) (Not Required)
- Week 4: Hibernate/JPA (Not Required)
- Week 5: Spring framework

Detailed day wise breakup:

Day 1:

- Introduction to Core Java Programming and Concepts
  - A First Look
  - A Simple Java Class
  - Java's "Hello World" Program
  - Java Basics
  - Language and Platform Features
  - Program Life Cycle
  - The Java SE Development Kit (JDK)
  - Working with the Development Environment
- Flow of Control
  - Branching: if, if-else, switch
  - Iteration: while, do-while, for, enhanced for, break, continue
- Introduction to Junit and unit testing
  - Understanding the test scenario
  - Test boundaries
  - What can be tested?
  - Creating test cases
  - Creating test suites

Day 2:

- Class and Object Basics
  - The Object Model and Object-Oriented Programming
  - Classes, References, and Instantiation
  - Garbage Collection
  - Adding Data to a Class Definition
  - Adding Methods (Behavior)
- Packages
  - Package Overview - Using Packages to Organize Code
  - import statements
  - Creating Packages, package Statement, Required Directory Structure

- Finding Classes, Packages and Classpath

- More on Classes and Objects
    - Accessing data, the "this" variable
    - Encapsulation and Access Control, public and private Access
    - Constructors and Initialization
    - static Members of a Class
    - Scopes, Blocks, References to Objects

Day 3:

- Composition and Inheritance
    - Using Composition to Deal With Complexity
    - Composition/HAS-A, Delegation
    - Using Inheritance and Polymorphism to share commonality
    - IS-A, extends, Inheriting Features, Overriding Methods, Using Polymorphism
    - Class Object
    - Abstract Classes

Day 4:

- Exceptions
    - Exceptions and the Exception Hierarchy
    - try, catch and finally
    - Handling Exceptions
    - Program Flow with Exceptions
    - Creating user defined exceptions and exception funneling

Day 5:

- Interfaces
    - Using Interfaces to Define Types
    - Interfaces and Abstract Classes
    - Default Methods and static Methods (Java 1.8 or later only)
    - Using Interfaces to Remove Implementation Dependencies

Day 6:

- Java Collections and Generics
    - The Collections Framework and its API
    - Collections and Java Generics
    - Collection, Set, List, Map, Iterator
    - Auto boxing
    - Collections of Object (non-generic)
    - Using ArrayList, HashSet, and HashMap
    - Processing items with an Iterator
    - More about generics

Day 7:

- I/O Streams
    - Readers and Writers
    - Filter Streams
    - Byte Streams
    - Formatted Output

Day 8:

- Database Access with JDBC
    - JDBC Overview
    - JDBC Architecture
    - Drivers and types of drivers
    - DriverManager,
    - Connection,
    - Statement, PreparedStatement, CallableStatement

Day 9:

- ResultSet
    - ResultSetMetaData
    - DatabaseMetaData

Day 10:

- Additional Java Features
    - Enums

  - Annotations
  - Lambda Expressions and Method References

Day 11:

- Spring overview and architecture
  - What is Spring framework?
  - Why Spring framework?
  - Spring framework architecture
  - Usage scenario
  - Step by step refactoring of Helloworld application using Dependency Injection
- Spring Dependency Injection Basics
  - What is and Why Dependency Injection (DI)?
  - Two DI variants
  - Reading configuration
  - Bean configuration
  - Bean parameter types
  - Auto-wiring and auto-scanning
  - Bean naming
- Spring Dependency Injection Annotation
  - Annotation-based Dependency Injection -@Autowired, @Required
    Qualifier - @Qualifier, Custom qualifier
  - JSR 330 (Dependency Injection for Java) - @Inject
  - JSR 250 (Common Annotations) -@PostConstruct & @PreDestroy, @Resource
    @Component and further stereotyped annotations - @Service, @Repository, @Controller
  - Auto scanning -@ComponentScan

Day 12:

- Spring Database Introduction
  - DAO support
  - @Repository annotation
  - Data access through ORM
- Spring AOP
  - What is and Why AOP?
  - AOP concepts and terminology
  - @AspectJ support in Spring
  - Types of advice
  - Declaring advices
  - Accessing Join point information
  - Declaring a pointcut
  - Defining and using common pointcuts

Day 13:

- Spring Transaction
  - Transaction management in Spring framework
  - Global transaction vs. local transaction
  - PlatformTransactionManager interface
  - Declarative transaction management
  - Transaction propagation
- Spring Data JPA
  - What is and Why Spring Data?
  - Spring Data JPA
  - Spring Data Repository interfaces
  - Step by step of building Spring Data JPA application
  - Paging and Sorting
  - Query generation strategies
- Spring Boot
  - What is and Why Spring Boot?
  - Getting started with Spring Boot
  - Building a Web app using Spring Boot
  - Auto-configuration
  - SpringApplication class

- External configuration
- Actuator
- Misc. features

- Spring 4 MVC Introduction
  - Introduction to Spring MVC
  - DispatchServlet, Context configuration
  - SpringMVC interfaces

Day 14:

- Spring 4 MVC Controllers Part I
  - What is a Controller?
  - Request mapping
  - Handler method arguments – Implicit models
  - Handler method return types (used for view selection)
- Spring 4 MVC Controllers Part II
  - URI template
  - Mapping requests with other means (in addition to URL)
  - Handler method arguments - @PathVariable, @RequestParam
  - Type conversion
  - Handler method that directly creates a HTTP response
  - Interceptor
  - Automatic attr. key name generation
- Spring 4 MVC View Resolvers
  - Resolving views
  - Spring-provided view resolvers
  - Chaining view resolvers
  - Views vs @ResponseBody
  - Automatic logical view name generation
  - ViewController & RedirectViewController
  - ContentNegotiatingViewResolver
- Spring 4 MVC Form handling
  - 2-phase form submission handling
  - Command/form objects
  - @ModelAttribute
  - Validation
  - Data binding
  - Form tags
  - Redirection (in form submission handling)

Day 15:

- Spring 4 MVC Misc.
  - Exception handling
  - Locale handling
  - @Value
  - SpEL
  - Static resource configuration
  - Logging
  - Debugging
- Spring 4 REST Design
  - Create Object models
  - Design URIs
  - Determine Data formats
  - Determine HTTP methods to use
  - Support HATEOAS
- Spring 4 MVC REST
  - CRUD operations via REST calls
  - REST client tools
  - @ResponseBody
  - Representations (Formats) - produces and consumes
  - XML binding
  - Create/Update/Delete

  - @RestController
- Spring REST using JAX-RS: Resource matching
  - Creating resources
  - @Path
  - HTTP method annotations (Uniform interface)
  - @GET, @POST, @PUT, @DELETE
  - Building REST application step by step
  - Sub-resource locator
- Spring REST using JAX-RS: Injection annotations
  - @PathParam, @MatrixParam, @QueryParam, @FormParam, @BeanParam
    @HeaderParam, @CookieParam
    @Context
- Spring REST using JAX-RS: Content Negotiation
  - HTTP media types
  - @Produces and @Consumes
  - Content handler
  - Content type selection by client
- Spring REST using JAX-RS: Response generation
  - Two schemes of generating responses
  - Creating responses using built-in Entity Provider (Content-handler)
  - Creating responses using ResponseBuilder
  - XML content handler via JAXB
  - JSON content handler
  - JAXBElement
  - Creating response for “Create” operation