نام دوره : دوره فشرده مقدماتی و پیشرفته جاوا 11

خلاصه دوره:

این دوره فشرده از دو قسمت مقدماتی و پیشرفته تشکیل شده است. در بخش مقدماتی به نحوه پیاده سازی مفاهیم شی گرا با استفاده از زبان جاوا پرداخته می شود سپس هسته اصلی و استاندارد زبان جاوا بهمراه بررسی قابلیت آن مورد بررسی قرار می گیرد. مباحثی همچون مدیریت خطا و استثناء ، مدیریت فایل ، ارتباط با بانک اطلاعات رابطه ای و اصول برنامه نویسی همروند در زبان جاوا ارائه می گردد. در بخش پیشرفته این دوره ویژگیهای اضافه شده در زبان جاوا در نسخه 11 مطرح می گردد سپس به ساختار معماری چند لایه برنامه های کاربردی وبی جاوایی پرداخته می شود. نحوه پیاده سازی برنامه کاربردی وبی و نحوه ارائه وب سرویس در این بخش ارائه می گردد.

اهداف دوره:

یادگیری نحوه پیاده سازی مفاهیم شی گرا در زبان جاوا

یادگیری هسته اصلی زبان جاوا

برنامه نویسی همروند با زبان جاوا

ویژگیهای اضافه شده در نسخه 11 زبان جاوا

یادگیری نحوه پیاده سازی برنامه های کاربردی وبی با معماری چند لایه به زبان جاوا

Java SE 11 and Java For Enterprise Capabilities Overview Package

Duration: 48 Hours

Required Prerequisites:

- Experience with Object-Oriented or Structural Programming Language such as C/C++ or C# Language.
- Familiar with Relational Database Concepts (such as Tables, Relations, SQL).
- Familiar with JSON and XML.

Course Overview

Java 11 Programming is the first part of this training and covers the core language features and Application Programming Interfaces (API) you will use to design object-oriented applications with Java SE 11 Platform. Developing Applications using Java for Enterprise is the second part of this training. The technologies will be presented in this course include annotations, Java Persistence API (JPA), Java Transaction API (JTA), Servlets, Contexts and Dependency Injection (CDI), JAX-RS RESTful and SOAP Web Services, Batch API, Timer services and Java Concurrency.

Java SE 11 Objectives

- Creating high-performing multi-threaded applications
- Creating Java technology applications that leverage the object-oriented features of the Java language, such as encapsulation, inheritance, and polymorphism
- Implementing input/output (I/O) functionality to read from and write to data and text files and understand advanced I/O streams
- Executing a Java technology application from the command line
- Manipulating files, directories and file systems using the JDK NIO.2 specification
- Creating applications that use the Java Collections framework
- Performing multiple operations on database tables, including creating, reading, updating and deleting using both JDBC and JPA technology
- Searching and filter collections using Lambda Expressions
- Implementing error-handling techniques using exception handling
- Multithreading
- Implementing Modularity

Java For Enterprise Objectives

- Defining how the Java language achieves platform independence
- Evaluating Java libraries, middle-ware, and database options
- Defining how the Java language continues to evolve
- Use Java Persistence and Java Transaction APIs.
- Provide Timer, Concurrency and Batch services.

- Create SOAP and REST Web Services.
- Assemble and deploy Java applications to a Java Application Server.

Detailed Java SE 11 Outline

- **Creating simple Java classes**
  - Creating primitive variables
  - Using operators
  - Creating and manipulate strings
  - Using if-else and switch statements
  - Iterating with loops: while,do-while,for,enhanced for
  - Creating arrays
  - Using Java fields, constructors, and methods
- **Encapsulation and Sub classing**
  - Using encapsulation in Java class design
  - Modeling business problems using Java classes
  - Making classes immutable
  - Creating and use Java subclasses
  - Overloading methods
- **Overriding Methods, Polymorphism, and Static Classes**
  - Using access levels: private, protected, default, and public.
  - Overriding methods
  - Using the instanceof operator to compare object types
  - Using upward and downward casts
  - Modeling business problems by using the static keyword
  - Implementing the singleton design pattern
- **Abstract and Nested Classes**
  - Designing general-purpose base classes by using abstract classes
  - Constructing abstract Java classes and subclasses
  - Applying final keyword in Java
  - Distinguish between top-level and nested classes
- **Interfaces and Lambda Expressions**
  - Defining a Java interface
  - Choosing between interface inheritance and class inheritance
  - Extending an interface
  - Defaulting methods
  - Anonymous inner classes
  - Defining a Lambda Expression
- **Collections and Generics**
  - Creating a custom generic class
  - Using the type inference diamond to create an object
  - Creating a collection by using generics
  - Ordering collections
- **Exceptions and Assertions**
  - Defining the purpose of Java exceptions
  - Using the try and throw statements

  - Using the catch, multi-catch, and finally clauses
  - Auto close resources with a try-with-resources statement
  - Recognizing common exception classes and categories
  - Creating custom exceptions
  - Testing invariants by using assertions
- **Java Date/Time API**
  - Creating and manage date-based events
  - Creating and manage time-based events
  - Combining date and time into a single object
  - Working with dates and times across time zones
  - Managing changes resulting from daylight savings
  - Defining and create timestamps, periods and durations
  - Applying formatting to local and zoned dates and times
- **I/O Fundamentals**
  - Describing the basics of input and output in Java
  - Read and write data from the console
  - Using streams to read and write files
  - Writing and read objects using serialization
- **File I/O (NIO.2)**
  - Using the Path interface to operate on file and directory paths
  - Using the Files class to check, delete, copy, or move a file or directory
  - Using Stream API with NIO2
- **Concurrency**
  - Describing operating system task scheduling
  - Creating worker threads using Runnable and Callable
  - Using an ExecutorService to concurrently execute tasks
  - Identifying potential threading problems
  - Using synchronized and concurrent atomic to manage atomicity
  - Using monitor locks to control the order of thread execution
  - Using the java.util.concurrent collections
- **Database Applications with JDBC**
  - Defining the layout of the JDBC API
  - Connecting to a database by using a JDBC driver
  - Submitting queries and get results from the database
  - Specifying JDBC driver information externally
  - Performing CRUD operations using the JDBC API
- **Localization**
  - Describing the advantages of localizing an application
  - Defining what a locale represents
  - Read and set the locale by using the Locale object
  - Building a resource bundle for each locale
  - Calling a resource bundle from an application
  - Changing the locale for a resource bundle
- **Java 11 New Features**
  - LTS Java Versions and Oracle JDK 11 vs OpenJDK 11
  - Modularity
  - --release flag
  - Multi-jar releases
  - Var keyword
  - Version String Schema

- Performance Enhancements
- New String Methods
- New File Methods
- Collection to an Array
- The Not Predicate Method
- Local-Variable Syntax for Lambda
- HTTP Client

Developing Applications using Java For Enterprise Outline

Introduction to Java For Enterprise Capabilities

- Standards, containers, APIs, and services
- Application component functionalities mapped to tiers and containers
- Interconnect Application Components with CDI Annotations and JNDI
- Web container technologies, Business logic implementation technologies, and web service technologies
- Packaging and deployment
- Enterprise JavaBeans, managed beans, and CDI beans
- Understanding lifecycle and memory scopes
- Linking components together with annotations, injections, and JNDI

Managing Persistence by Using JPA Entities

- Create JPA entities with Object-Relational Mappings (ORM)
- Use Entity Manager to perform database operations with JPA entities
- Handle entity data with conversions, validations, and key generation
- Describe persistence management and locking mechanisms
- Create and execute JPQL statements

Implementing SOAP Services by Using JAX-WS

- Describe a SOAP Web Service structure
- Create SOAP Web Services using JAX-WS API
- Create SOAP Web Service clients

Creating Java Web Applications by Using Servlets, Interceptors and Filters

- Describe HTTP basics
- Create Java servlet classes and map then to URLs
- Handle HTTP headers, parameters, cookies
- Use servlets to handle different content types
- Manage servlet life cycle with container callback methods
- Use CDI Managed Beans
- Use Interceptors and Filters
- Implement asynchronous servlets and use NIO API

Implementing REST Services using JAX-RS API

- Understand REST service conventions
- Create REST services using JAX-RS API
- Consume REST service within the client tier

How to convert Java object to / from JSON

- Jackson
- Gson