Java SE 8 Programing

Course Specifications

**Course Length:**

5 days

Course Description

**Overview:**

This course covers the core APIs that you use to design object-oriented applications with Java. This course also covers writing database programs with JDBC. Use this course to further develop your skills with the Java language

**Skillset:**

Intermediate Skillset

**Course Objectives:**

 You will:

* Create Java technology applications that leverage the object-oriented features of the Java language
* Execute a Java application from the command line
* Create applications that use the Collections framework
* Search and filter collections by using Lambda Expressions
* Implement error-handling techniques by using exception handling
* Implement input/output (I/O) functionality to read from and write to data and text files and understand advanced I/O streams
* Manipulate files, directories, and file systems by using the NIO.2 specification
* Perform multiple operations on database tables, including creating, reading, updating, and deleting, by using the JDBC API
* Create high-performing multithreaded applications that avoid deadlock
* Use Lambda Expression concurrency features

**Target Audience:**

Java programmers with intermediate or advanced skills

**Suggested Prerequisites:**

Course-specific Technical Requirements

**Virtual Delivery Option:**

* Computer or Laptop with sufficient capabilities to connect via virtual platform and operate efficiently
* If requesting pdf version of the courseware and labs, dual monitors are
* Quiet uninterrupted environment to allow for dedicated learning experience

**Software:**

* Chrome browser (latest updated version preferred)

Course Content

**Module 01: Basic Java Overview**

* Java Server Environments
* Java Community
* Internet of Things
* Java Community Process

**Module 02: Java Syntax**

* Create simple JAVA Classes
* Manage Control Flow
* Use Java fields
* Use package and Import methods
* Lab Creating Java Classes

**Module 03: Encapsulation**

* Use encapsulation in Java class design
* Model business problems by using Java classes
* Make classes immutable
* Create and use Java subclasses
* Overload methods
* Lab Encapsulation

**Module 04: Overriding Methods, Polymorphism, and Static Classes**

* Use access levels: private, protected, default, and public
* Override methods
* Use virtual method invocation
* Use varargs to specify variable arguments
* Use the instanceof operator to compare object types
* Use upward and downward casts
* Model business problems by using the static keyword
* Implement the singleton design pattern
* Lab Overriding Methods

**Module 05: Abstract and Nested Classes**

* Design general-purpose base classes by using abstract classes
* Construct abstract Java classes and subclasses
* Apply the final keyword in Java
* Distinguish between top-level and nested classes
* Lab Abstract classes

**Module 06: Interfaces and Lambda Expressions**

* Define a Java interface
* Choose between interface inheritance and class inheritance
* Extend an interface
* Define a lambda expression
* Lab Interfaces

**Module 07: Generics and Collections**

* Create a custom generic class
* Use the type inference diamond to create an object
* Create a collection without using generics
* Create a collection by using generics
* Implement an ArrayList
* Implement a TreeSet
* Implement a HashMap
* Implement a Deque
* Order collections
* Lab Collections

**Module 08: Collections, Streams, and Filters**

* Describe the Builder pattern
* Iterate through a collection by using lambda syntax
* Describe the Stream interface
* Filter a collection by using lambda expressions
* Call an existing method by using a method reference
* Chain multiple methods
* Define pipelines in terms of lambdas and collections
* Lab Streams and Filters

**Module 09: Lambda Built-in Functional Interfaces**

* List the built-in interfaces included in java.util.function
* Use primitive versions of base interfaces
* Use binary versions of base interfaces
* Lab Lamdba Functions

**Module 10: Lamdba Operations**

* Extract data from an object by using map
* Describe the types of stream operations
* Describe the Optional class
* Describe lazy processing
* Sort a stream
* Save results to a collection by using the collect method
* Group and partition data by using the Collectors class
* Lab Lamdba Operations

**Module 11: Exceptions and Assertions**

* Define the purpose of Java exceptions
* Use the try and throw statements
* Use the catch, multi-catch, and finally clauses
* Autoclose resources with a try-with-resources statement
* Recognize common exception classes and categories
* Create custom exceptions and auto-closeable resources
* Test invariants by using assertions
* Lab Expressions and assertions

**Module 12: Java Date/Time API**

* Create and manage date-based events
* Create and manage time-based events
* Combine date and time into a single object
* Work with dates and times across time zones
* Manage changes resulting from daylight savings
* Define and create timestamps, periods, and durations
* Apply formatting to local and zoned dates and times
* Lab Java Date/Time

**Module 13: Java I/O Fundamentals**

* Describe the basics of input and output in Java
* Read data from and write data to the console
* Use I/O streams to read and write files
* Read and write objects by using serialization
* Lab Java I/O

**Module 14: Java File I/O (NIO.2)**

* Use the Path interface to operate on file and directory paths
* Use the Files class to check, delete, copy, or move a file or directory
* Use Stream API with NIO2
* Lab Java File I/O

**Module 15: Concurrency**

* Describe operating system task scheduling
* Create worker threads using Runnable and Callable
* Use an ExecutorService to concurrently execute tasks
* Identify potential threading problems
* Use synchronized and concurrent atomic to manage atomicity
* Use monitor locks to control the order of thread execution
* Use the java.util.concurrent collections
* Lab Concurrency

**Module 16: The Fork-Join Framework**

* Apply the Fork-Join framework
* Lab Fork-Join Framework

**Module 17: Parallel Streams**

* Review the key characteristics of streams
* Contrast old style loop operations with streams
* Describe how to make a stream pipeline execute in parallel
* List the key assumptions needed to use a parallel pipeline
* Define reduction
* Describe why reduction requires an associative function
* Calculate a value using reduce
* Describe the process for decomposing and then merging work
* List the key performance considerations for parallel streams
* Lab Streams

**Module 18: Building Database Applications with JDBC**

* Define the layout of the JDBC API
* Connect to a database by using a JDBC driver
* Submit queries and get results from the database
* Specify JDBC driver information externally
* Perform CRUD operations by using the JDBC API
* Lab Database Applications

**Module 19: Localization**

* Describe the advantages of localizing an application
* Define what a locale represents
* Read and set the locale by using the Locale object
* Create and read a Properties file
* Build a resource bundle for each locale
* Call a resource bundle from an application
* Change the locale for a resource bundle
* Lab Database Applications