Advanced Java 8 Using Eclipse

Duration: **5 days**

Overview:

This course teaches how to develop advanced Java applications using Eclipse. The advanced features of Java that developers may be using in many different types of programs are covered. This course also covers many of the advanced features added in Java 7 and Java 8.

Target Audience:

This course is intended for programmers who are familiar with Java and want to learn about the advanced features of Java.

Pre-requisites:

Before attending this course, students must have a good understanding of object-oriented programming using Java.

MODULE 1: Java Logging API

- Java Logging API
- Control Flow of Logging
- Logging Levels
- Logging Handlers
- Loggers
- Logging Example
- Logging Formatters & Log Manager
- Logging Configuration File
- Example Logging Configuration File
- Logging Filters

MODULE 2: JDBC

- Overview
- Types of Drivers
- Making a Connection
- Statements
- Execute, Update and ResultSets
- JDBC and try-with-resources
- SQLException
- Prepared Statements
- Stored Procedures
- Transactions
- Connection Pooling

MODULE 3: Overview of Java Persistence API

- Data Persistence
- Java Persistence API 2.0
- Entities
- Session EJB vs JPA Entities
- Entities
- Persisting and Retrieving Data
- Accessing Entities
- EntityManager & Persistence Unit
- Persistence Context
- Entities - Example
- persistence.xml – Hibernate Provider
- persistence.xml – Open JPA Provider
- persistence.xml - Toplink
- Entity Instance Lifecycle
- Creating EntityManager in Session EJB
- Creating EntityManager in a Plain Java Class
- Working With the EntityManager Interface
- Transaction Basics
- Summary

MODULE 4: The Java Architecture for XML Binding (JAXB)

MODULE 5: Networking

- Overview
- URL Connections
- Browser Example
- InetAddress
- Socket Classes
- Simple Clients and Servers
- Simple Clients and Servers (cont.)
- Multithreaded Servers
- UDP Sockets

MODULE 6: Internationalisation

- Internationalization
- Adoption Stages
- Internationalization
- Locale
- Dates
- User Interface design
- Resource Bundles
- Other Local Customs
- How Java Represents Characters
- Text Files
- Text files
- Summary

MODULE 7: Using the Date/Time API

- Introduction to the Date/Time API
- Create Date Events
- Create Time Events
- Working with Date and Time Together
- Working with Time Zones
- Working with Durations

MODULE 8: Annotations

- The Annotations Model
- Annotation Types and Annotations
- Built-In Annotations
- JSR 250 Common Annotations
- JSR 250 Common Annotations Example
- Meta-Annotations
- Annotations vs. Descriptors (XML)
- Aspect-Oriented Programming (AOP)
- Aspect-Oriented Programming and Java
- @AspectJ Annotations Support

MODULE 9: Security

- Overview of JDK Security Features
- Java Cryptography Architecture (JCA)
- Java Cryptography Extension
- Using the MessageDigest Class
- Example of Using the MessageDigest Class
- Using the Signature Class
- Java Security Architecture
- Security Model – Sandbox
- Security Model – Trusted Signed Code & Security Policy
- JDK 1.4 Security Enhancement
- Protection Domains and Security Policies
- ProtectionDomain Class
- Permission Classes
- Using Permission Classes
- Policy Class
- Policy Configuration File
- AccessController Class
- SecurityManager Class
- Using the SecurityManager Class
- Java Authentication and Authorization Service - JAAS
- JAAS – Common Classes
- JAAS - Authentication
- JAAS – Authentication Configuration
- JAAS - Authorization
- Java Security Tools
- Using Java Security Tools – Code Signing
- Summary

MODULE 10: Java NIO and NIO.2

- NIO and NIO.2 Overview
- The java.nio.file.Path Interface
- Obtaining a Path Instance
- Path Operations
- Converting Paths
- Operations With Two Paths
- Working With Files
- File Attributes
- Working With File Attributes
- FileVisitor API
- Finding Files
- Watching Directories
- WatchService Example
- Buffers
- Channels
- Using Buffers and Channels - Write Example
- Using Buffers and Channels - Read Example
- Working With Legacy java.io.File Code
- Summary

MODULE 11: Threads

- Overview of Threads
- Threads in Java Programming
- Write a Runnable Class
- Create Threads
- Another Way of Creating Threads
- Two ways of creating threads
- States in a Thread's Lifetime
- JVM Scheduler
- Control and Schedule Thread
- Coordinating the Concurrency of Multiple threads
- Synchronization
- How Does the Object Lock Flag Work
- Using the synchronized keyword
- The Implication of Synchronization
- Example of Synchronization - Producer/Consumer
- Example of Synchronization – MyStack
- Example of Synchronization – Producer
- Example of Synchronization – Consumer
- Example of Synchronization – SyncTest
- Why Coordination is Required
- Coordinating Thread Cooperation
- wait() and notify()
- Example of Coordination Producer/Consumer
- Example of Coordination – MyStack
- Results
- Deadlock
- Method References

MODULE 12: Java Concurrency

- Java Concurrency
- Executor Interface
- Using the Executor
- Callable Interface
- Callable Example
- ExecutorService Interface
- Future Object
- Using Executor, Future and Callable
- Atomic Variables
- Using Atomic Variables
- Summary

MODULE 13: Fork/ Join Framework

- Fork/Join Introduction
- Fork Join Tasks
- RecursiveTask
- RecursiveTask Example
- RecursiveAction
- ForkJoinPool
- Summary

MODULE 14: Introduction to Lambda Expressions

- Purpose of Lambda Expressions
- Functional Interfaces
- Comparison to Anonymous Classes
- Syntax

MODULE 15: Collections Stream API

- Iterating Through a Collection with forEach
- Stream Interface
- Filtering with Lambda Expressions
- Using the map Method to Extract Data

MODULE 16: Using Built-In Lambda Types

- Built-in Interfaces of java.util.function Package
- Determining true or false with a Predicate
- Processing one Object and Return Another with Function
- Processing an Object and Return Nothing with Consumer
- Generating a New Object with Supplier

MODULE 17: Advanced Functional Programming

- Searching for Data Using Search Methods
- Sorting a Stream
- Making a Stream Pipeline Execute in Parallel
- Calculating a Value Using Reduction
- Modifying and Updating a Collection
- List, Walk, and Search a Tree Structure
- Flatten a Stream Using flatMap

MODULE 18: JUnit

- What is JUnit?
- Why JUnit?
- The xUnit Philosophy
- Test-Driven Design
- A JUnit Test
- Running the Tests
- Swing-based Test Runner
- Text-based Test Runner
- JUnit Basics
- assertTrue
- assertEquals
- assertSame
- assertNull
- The Failure Message
- The Test Class
- The Test Method
- The Test Suite
- JUnit with Annotations
- JUnit 4 Test Suite
- JUnit Design
- Testing Strategies
- Specific Techniques
- Testing Simple Java classes
- Testing with Databases
- Testing Web Applications
- Testing Java EE Web Applications
- JUnit with Ant
- Summary
- JUnit with Eclipse
- Create a Test Case
- Test Case "Stubs"
- Running Tests
- Eclipse Test Runner Icons
- Rerun an Individual Test
- Failure Trace
- Debug with JUnit
- Test Suite Wizard

MODULE 19: Mockito

- The Problem
- Old Solutions
- Bad Solutions?
- What's the other choice?
- Choices
- Mocking stubs
- Mocking Mocks
- Mostly Done
- Other Features
- Spy
- Annotations
- Summary

MODULE 20: Summary of Recent Java Changes

- Java 7 – Major New Features
- Java 7 – Generic Diamond Operator
- Java 7 – Catching Multiple Exceptions
- Java 7 – Rethrowing Exceptions
- Java 7 – try-with-resources Statement
- Java 7 – Suppressed Exceptions in try-with-resources
- Java 7 – Strings in switch Statement
- Java 7 – Changes in Numeric Literals
- Java 7 – Fork & Join Parallel Processing
- Java 7 – NIO.2 File Systems
- Java 8 – Major New Features
- Java 8 – Lambda Expressions
- Java 8 – Default Methods
- Java 8 – Collections Stream API
- Java 8 – Date & Time API
- Java 8 – Concurrency Changes
- Java 8 – Nashorn JavaScript Engine
- Java 8 – Repeating Annotations
- Java 8 – Type Annotations
- Java 8 – Security Changes
- Java 8 – HotSpot JRE Changes
- Summary

MODULE 21: Appendix A. Parsing XML with SAX

- SAX
- How it Works
- Core SAX2 Handler Classes
- SAX2 DefaultHandler
- SAX Events
- Ignorable Whitespace
- Parsing a Document
- Using SAXParserFactory
- Parse XML with SAX – Details
- Define an Event Handler
- Create a SAXParserFactory instance
- Define an Event Handler – startElement()
- Define an Event Handler – Element Attributes
- Define an Event Handler – Get Number of Attributes
- Define an Event Handler – Get Name of Attributes
- Define an Event Handler – Get Attribute Values
- Define an Event Handler – An Example
- Define an Event Handler – characters()
- Using characters()
- Define an Event Handler – Error Handling
- Define an Event Handler – ErrorHandler interface
- Parse XML Document
- Simple SAX Parser
- EntityResolver
- Locator
- Document Locator

MODULE 22: Appendix B. Parsing XML With Dom

- DOM
- Limitations of SAX
- XML as an Object Model
- Nodes
- The Basic Node Types
- Less Common Node Types
- Node Interface
- Document Interface
- NodeList Interface
- Element Interface
- Attr Interface
- Text Interface
- DOM Parsing
- Parse XML with DOM – Steps
- Prepare DOM Parser Object
- Parse XML Document
- Parse Exceptions
- Example – SimpleDOMParser
- Writing DOM