# Java

## Duration: 5 Days

## Software requirements:

* Java JDK 1.8
* Eclipse Mars
* MySQL database
* Acrobat Reader for viewing case studies and assignments

## Course Contents

### Day 1

**Introduction**

* Features of Java
* Java Architecture
* Java Runtime Environment, Class loaders, Java Virtual Machine and JDK

**Object Orientation**

* Thinking in Objects
* Class
* Class Specification
* Characteristics of an Object
* Basic Principles of Object Orientation
  + Abstraction
  + Encapsulation
  + Modularity
* Constructors
* Relationships between Classes
* Static attributes and Static Methods
* Packaging in Java

1. **Programming in Java** 
   * Where storage lives
   * Special case: primitive types
   * Scoping and Garbage Collection
   * Static and Instance initializers
   * Generalization –Specialization relationship
   * Method overloading and overriding

* Up casting and down casting (instance of)
* Abstract Classes
* Preventing Inheritance and the final keyword
* Protecting Access
* Object class and the singly rooted Hierarchy
* The Class called Class (Run- Time Type Identification)

### Day 2

**Interfaces**

* What is an interface?
* Establishing Realization relationship
* Writing loosely coupled applications using Interface

**Inner Classes**

* Static and Non-static Inner classes
* Anonymous classes

**Reflection API**

* The Class class
* The java.lang.reflect Package
* Reading Type Information
* Navigating Inheritance Trees
* Dynamic Instantiation
* Dynamic Invocation

**Annotations**

* Types of Annotation
  1. Marker
  2. Single element
  3. Multi value
* Meta Annotations
  1. Target
  2. Retention
  3. Documented

**Generics**

* Writing parameterized classes
* Defining bounds for typed variable
* Effective use of generics and code reuse

**Enum**

Use enum for constants

Use enum for creating immutable objects

### Day 3

**Error Handling with Exceptions**

* Basic exceptions
* Exception arguments
* Catching an exception
* The try block
* Exception handlers
* Checked and unchecked exceptions
* Creating your own exceptions
* Catching any exception
* Re -throwing an exception
* Standard Java exceptions
* Usage of try-with-resources Statement
* Exception guidelines

Collections

* Advanced Array Operations

Copying Arrays Comparing Arrays

* 1. The Array Class

Sorting arrays and the Equals Implementation Containers

Filling Containers

* Container Taxanomies
* Collection Functionality
* List Functionality
* Disadvantages of Object Containers
* Making a type-conscious ArrayList
* Parameterized types and Generic Collections
* Iterators
* Stack and Queue Implementations
* Set Functionality
* Sorted Set
* Map functionality

**Threads**

* Describe a thread
* Create separate threads, controlling the code and data that are used by that thread
* Use the keyword synchronized to protect data from corrupt ion
* Use wait() and notify() to communicate between threads
* Multithreaded Programs
* Synchronizing Threads

### Day 4

**Input/Output**

* Byte Streams vs. Character Streams
* Java Stream Hierarchy
* Chaining of Streams and adding functionality
* Standard Input: Reading strings
* Standard Input: Reading Primitives
* Writing to the Standard Output
* File I/O
* Other java .io Classes
* Converting data
* Object serialization
* Controlling serialization using Externalizable
* The transient keyword
* Versioning

**Java JSON APIs**

* What is JSON?
* Jackson- ObjectMapper
* Reading JSON from a String
* Reading JSON from a Reader
* Reading JSON from a File
* Reading JSON from a URL
* Reading JSON from a InputStream
* Generate JSON from objects
* Jackson Annotations

**Java XML Parsing**

* Java DOM Parser
* Java SAX Parser
* DOM4J Parser

**JDBC**

* Introduction
* DB Drivers
* Driver Manager
* Connection
* Statements and prepared statements
* Result sets and Meta Data

### Day 5

**Java 8 Lambda expressions**

* Lambdas: intro, motivation, and big idea
* Lambda expression interpretation
* Most basic form of lambdas
* Type inferencing
* Expression for body
* Comparing Java and JavaScript sorting
* Omitting parens
* Using effectively final variables
* @FunctionalInterface
* Method references
* java.util.function package
* Lambda building blocks in java.util.function: overview
* Predicate
* Function and BiFunction
* Consumer
* Supplier
* BinaryOperator
* Variable scoping: Local variables, instance variables, "this"
* Method references: details -- Class::staticMethod, variable::instanceMethod Class::instanceMethod, Class::new
* New features in Java 8 interfaces: Default methods (also called "virtual extension methods" or "defender methods") and static methods
* Methods that return lambdas: From Predicate: and, or, negate, isEqual; From Function: andThen, compose, identity; From Consumer: chain; Custom methods

**Java 8 Streams**

* Overview of Streams
* Building Streams
* Outputting Streams into arrays or Lists
* Core Stream methods: overview, forEach, map, filter, findFirst
* Lazy evaluation and short-circuit operations
* Parallel Streams
* Infinite Streams (really unbounded streams with values that are calculated on the fly)
* **Reactive Programming with Java 8 using RxJava**
  + Creating and Connecting Observables, Observers, and Subjects
    - The Observable.from method
    - The Observable.just method
    - Other Observable factory methods
    - The Observable.create method
    - Subscribing and unsubscribing
    - Hot and cold Observable instances The ConnectableObservable class
  + Transforming, Filtering, and Accumulating Your Data
    - Observable transformations Transformations with the various flatMap operators
    - Grouping items

Note: the following patterns will be covered as and when the topics are covered and not as separate topic on design pattern:

* Singleton
* Factory
* Iterator
* Observer