# Core Java 8

Java SE 8

Capgemini

# Course Goals and Non Goals

## ➢Course Goals

- Implementing OOPs features in Java
- Developing Java Desktop Applications
- Testing using Junit 4
- Implementing Multithreading

## ➢Course Non Goals

- Developing GUI applications

# Pre-requisites

- Basic Programming Concepts
- OOP

# Intended Audience

Developers new to Java technology

# Day Wise Schedule

## Day 1:

- Lesson 1: Declarations and Access Control
- Lesson 2: Object Orientation

## Day 2:

- Lesson 3: Assignments
- Lesson 4: Operators

## Day 3 :

- Lesson 5: Flow Control, Exceptions

## Day 4:

- Lesson 6: Maven Fundamentals
- Lesson 7 :TDD using junit 5

# Day Wise Schedule

## Day 5:

- Lesson 7: TDD using junit 5
- Lesson 8: Strings ,I/O Formatting and Parsing

## Day 6:

- Lesson 9:Generics And Collections

## Day 7:

- Lesson 10: Threads
- OCJP Preparation

## Day 8:

- Lesson 11: Concurrent Patterns in Java
- OCJP Preparation

## Day 9 :

- Lesson 12: Concurrent Collections

## Day 10

- Lesson 13: Lambda Expression
- Lesson 14: Stream API

## Day 11

- Lesson 15 : Layered Architecture

## Day 12 and 13

Parallel project orientation and Implementation

## Day 14 :

Parallel Project and OCJP

# Table of Contents

## Lesson 1: Declarations And Access Control

- Identifiers & JavaBeans
- Legal Identifiers
- Sun's Java Code Conventions
- JavaBeans Standards
- Declare Classes
- Source File Declaration Rules
- Class Declarations and Modifiers
- Concrete Subclass
- Declaring an Interface
- Declaring Interface Constants
- Declare Class Members
- Access Modifiers
- Non-access Member Modifiers
- Constructor Declarations
- Variable Declarations
- Declaring Enums

# Table of Contents

## Lesson 2: Object Orientation

- Encapsulation
- Inheritance, Is-A, Has-A
- Polymorphism
- Overridden Methods
- Overloaded Methods
- Reference Variable Casting
- Implementing an Interface
- Legal Return Types
- Return Type Declarations
- Returning a Value
- Constructors and Instantiation
- Default Constructor
- Overloaded Constructors
- Statics
- Static Variables and Methods
- Coupling and Cohesion

# Table of Contents

## Lesson 3: Assignments

- Stack and Heap—Quick Review
- Literals, Assignments, and Variables
- Assignment Operators
- Casting Primitives
- Using a Variable or Array Element That Is Uninitialized and Unassigned
- Local (Stack, Automatic) Primitives and Objects
- Passing Variables into Methods
- Passing Object Reference Variables
- Does Java Use Pass-By-Value Semantics?
- Passing Primitive Variables
- Array Declaration, Construction, and Initialization
- Initializing Blocks
- Using Wrapper Classes and Boxing
- An Overview of the Wrapper Classes
- Creating Wrapper Objects
- Using Wrapper Conversion Utilities
- AutoBoxing
- Overloading
- Garbage Collection and memory Management
- Writing Code That Explicitly Makes Objects Eligible for Garbage Collection

# Table of Contents

## Lesson 4: Operators

- Java Operators
- Assignment Operators
- Relational Operators
- instanceof Comparison
- Arithmetic Operators
- Conditional Operator
- Logical Operators

## Lesson 5 : Flow Control ,Exceptions

- if and switch Statements
- if-else Branching
- switch Statements
- Loops and Iterators
- Using while Loops
- Using do Loops
- Using for Loops
- Using break and continue
- Unlabeled Statements
- Labeled Statements
- Handling Exceptions
- Catching an Exception Using try and catch
- Using finally
- Propagating Uncaught Exceptions

# Table of Contents

## Lesson 5: Continues…

- Defining Exceptions
- Exception Hierarchy
- Handling an Entire Class Hierarchy of Exceptions
- Exception Matching
- Exception Declaration and the Public Interface
- Rethrowing the Same Exception
- Common Exceptions and Errors

## Lesson 6 : Maven Fundamentals

- Introduction
- Folder Structure
- The pom.xml
- Dependencies
- Goals
- Scopes
- The Compiler Plugin
- Source Plugin
- Jar Plugin

# Table of Contents

## Lesson 7: TDD with Junit 5

- Types of Tests
- Why Unit Tests Are Important
- What's JUnit?
- JUnit 5 Architecture
- IDEs and Build Tool Support
- Setting up JUnit with Maven
- Lifecycle Methods
- Test Hierarchies
- Assertions
- Disabling Tests
- Assumptions
- Test Interfaces and Default Methods
- Repeating Tests
- Dynamic Tests
- Parameterized Tests
- Argument Sources
- TDD Introduction
- Types of Testing
- Testing Frameworks and Tools
- Testing Concepts,Mockito

# Table of Contents

## Lesson 8 : Strings, I/O Formatting and Parsing

- String, StringBuilder, and StringBuffer
- The String Class
- Important Facts About Strings and Memory
- Important Methods in the String Class
- The StringBuffer and StringBuilder Classes
- Important Methods in the StringBuffer and StringBuilder Classes
- File Navigation and I/O
- Types of Streams
- The Byte-stream I/O hierarchy
- Character Stream Hierarchy
- RandomAccessFile class
- The java.io.Console Class
- Serialization
- Dates, Numbers, and Currency
- Working with Dates, Numbers, and Currencies
- Parsing, Tokenizing, and Formatting
- Locating Data via Pattern Matching
- Tokenizing

# Table of Contents

## Lesson 9: Generics And Collections

- Overriding hashCode() and equals()
- Collections
- So What Do You Do with a Collection?
- List Interface
- Set Interface
- Map Interface
- Queue Interface
- Using the Collections Framework
- ArrayList Basics
- Autoboxing with Collections
- Sorting Collections and Arrays
- Navigating (Searching) TreeSets and TreeMaps
- Other Navigation Methods
- Backed Collections
- Generic Types
- Generics and Legacy Code
- Mixing Generic and Non-generic Collections
- Polymorphism and Generics

# Table of Contents

## Lesson 10:Threads

- Defining, Instantiating, and Starting Threads
- Defining a Thread
- Instantiating a Thread
- Starting a Thread
- Thread States and Transitions
- Thread States
- Preventing Thread Execution
- Sleeping
- Thread Priorities and yield( )
- Synchronizing Code
- Synchronization and Locks
- Thread Deadlock
- Thread Interaction
- Using notifyAll( ) When Many Threads May Be Waiting

# Table of Contents

## Lesson 11: Concurrent Patterns in Java

- Introducing Executors, What Is Wrong with the Runnable Pattern?
- Defining the Executor Pattern: A New Pattern to Launch Threads
- Defining the Executor Service Pattern, a First Simple Example
- Comparing the Runnable and the Executor Service Patterns
- Understanding the Waiting Queue of the Executor Service
- Wrapping-up the Executor Service Pattern
- From Runnable to Callable: What Is Wrong with Runnables?
- Defining a New Model for Tasks That Return Objects
- Introducing the Callable Interface to Model Tasks
- Introducing the Future Object to Transmit Objects Between Threads
- Wrapping-up Callables and Futures, Handling Exceptions

# Table of Contents

## Lesson 12: Concurrent Collections in Java

# Table of Contents

## Lesson 13 : Lambda Expression

- Introduction
- Writing Lambda Expressions
- Functional Interfaces
- Types of Functional Interfaces
- Method reference

## Lesson 14: Stream API

- Introduction
- Stream API with Collections
- Stream Operations

## Lesson 15: Layered Architecture

- Understand the concept of Layered Architecture
- Implement layers in Java applications

# References

Books:

- Java, The Complete Reference; by Herbert Schildt
- Thinking in Java; by Bruce Eckel
- Beginning Java 8 Fundamentals by Kishori Sharan

Websites:

- Java home page: http://java.sun.com/
- JDK 1.8 documentation: http://docs.oracle.com/javase/8/docs/
- Multithreading : https://docs.oracle.com/javase/tutorial/essential/concurrency/index.html

# Next Step Courses

Servlets

JSP

# Other Parallel Technology Areas

C ++

C#.Net

Visual Basic.Net