

BSc IT Department
Java Full Stack Development (Short Term Course)
Syllabus Academic Year 2022-23

Core Java 8

Program Duration: 12.5 days

Contents:

☐ **Declarations and Access Control**

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

☐

Object Orientation

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

☐

Assignments

- o Stack and Heap—Quick Review
- o Literals, Assignments, and Variables
- o Literal Values for All Primitive Types
- o Assignment Operators

- o Casting Primitives
- o Using a Variable or Array Element That Is Uninitialized and Unassigned
- o Local (Stack, Automatic) Primitives and Objects
- o Passing Variables into Methods
- o Passing Object Reference Variables
- o Does Java Use Pass-By-Value Semantics?
- o Passing Primitive Variables
- o Array Declaration, Construction, and Initialization
- o Declaring an Array
- o Constructing an Array
- o Initializing an Array
- o Initialization Blocks
- o Using Wrapper Classes and Boxing
- o An Overview of the Wrapper Classes
- o Creating Wrapper Objects
- o Using Wrapper Conversion Utilities
- o Autoboxing
- o Overloading
- o Garbage Collection
- o Overview of Memory Management and Garbage Collection
- o Overview of Java's Garbage Collector
- o Writing Code That Explicitly Makes Objects Eligible for Garbage Collection

□

Operators

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

□

Flow Control, Exceptions

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

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



Gradle Fundamentals

- o Introduction
- o Folder Structure
- o Install and Setup Gradle on Windows
- o Dependencies in Build Scripts
- o Gradle Wrapper
- o Lifecycle Tasks: The Base Plug In
- o Using Project Info and the check command
- o Creating Variables and external properties
- o Creating a Build Scan
- o Dependencies



TDD with Junit 5

- o Types of Tests
- o Why Unit Tests Are Important
- o What's JUnit?
- o JUnit 5 Architecture
- o IDEs and Build Tool Support
- o Setting up JUnit with Maven
- o Lifecycle Methods
- o Test Hierarchies
- o Assertions
- o Disabling Tests
- o Assumptions
- o Test Interfaces and Default Methods
- o Repeating Tests
- o Dynamic Tests
- o Parameterized Tests
- o Argument Sources
- o Argument Conversion
- o What Is TDD?
- o History of TDD
- o Why Practice TDD?
- o Types of Testing
- o Testing Frameworks and Tools
- o Testing Concepts
- o Insights from Testing
- o Mocking Concepts
- o Mockito Overview
- o Mockito Demo
- o Creating Mock Instances
- o Stubbing Method Calls

Strings, I/O, Formatting, and Parsing

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

□

Generics and Collections

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

□

Threads

- o Defining, Instantiating, and Starting Threads
- o Defining a Thread
- o Instantiating a Thread
- o Starting a Thread
- o Thread States and Transitions

- o Thread States
- o Preventing Thread Execution
- o Sleeping
- o Thread Priorities and yield()
- o Synchronizing Code
- o Synchronization and Locks
- o Thread Deadlock
- o Thread Interaction
- o Using notifyAll() When Many Threads May Be Waiting

Concurrent Patterns in Java

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

□

Concurrent Collections

- o Implementing Concurrency at the API Level
- o Hierarchy of Collection and Map, Concurrent Interfaces
- o What Does It Mean for an Interface to Be Concurrent?
- o Why You Should Avoid Vectors and Stacks
- o Understanding Copy On Write Arrays
- o Introducing Queue and Deque, and Their Implementations
- o Understanding How Queue Works in a Concurrent Environment
- o Adding Elements to a Queue That Is Full: How Can It Fail?
- o Understanding Error Handling in Queue and Deque
- o Introducing Concurrent Maps and Their Implementations
- o Atomic Operations Defined by the ConcurrentHashMap Interface
- o Understanding Concurrency for a HashMap
- o Understanding the Structure of the ConcurrentHashMap from Java 7
- o Introducing the Java 8 ConcurrentHashMap and Its Parallel Methods
- o Parallel Search on a Java 8 ConcurrentHashMap
- o Parallel Map / Reduce on a Java 8 ConcurrentHashMap
- o Parallel ForEach on a Java 8 ConcurrentHashMap
- o Creating a Concurrent Set on a Java 8 ConcurrentHashMap
- o Introducing Skip Lists to Implement ConcurrentHashMap
- o Understanding How Linked Lists Can Be Improved by Skip Lists
- o How to Make a Skip List Concurrent Without Synchronization

□ **Lambda Expressions**

- o Introduction
- o Writing Lambda Expressions

- o Functional Interfaces
 - o Types of Functional Interfaces
 - o Method reference
-

Stream API

- o Introduction
- o Stream API with Collections
- o Stream Operations

Introduction to Design Pattern

Self learning with online links and explanation by Trainer with Demos

- o Creational Design Pattern
 - Factory Pattern
 - Singleton Pattern
 - Prototype Pattern
- o Structural Design Pattern
 - Decorator Pattern
 - Facade Pattern
- o Behavioral Design Pattern
 - Chain of Responsibility Pattern
 - Iterator Pattern
- o Presentation Layer Design Pattern
 - Intercepting Filter Pattern
 - Front Controller Pattern
- o Business Layer Design Pattern
 - Business Delegate Pattern
 - Transfer Object Pattern
- o Integration Layer Design Pattern
 - Data Access Object Pattern

• DevOps (Git, SonarQube, Maven, Jenkins)

□ Introduction to DevOps

- o Introduction of DevOps
- o Dev And Ops
- o Agile Vs DevOps
- o Continuous Integration & Delivery pipeline
- o Tools For DevOps
- o Use-case walkthrough

□ GIT Hub

- o Working locally with GIT
- o Working remotely with GIT
- o Branching, merging & rebasing with GIT
- o Use Case walkthrough

- Jenkins:
 - o Introduction to Jenkins
 - o Jenkins Objective
 - o Introduction to continuous integration deployment & Jenkins-ci
 - o Continuous Deployment & distribution builds with Jenkins
- Sonar
 - o Introduction to Sonar
 - o Code quality Monitoring- Sonar
 - o Use Case walkthrough

Database Using

PostgreSQLDuration : 2

days

Contents:

□

Introduction

- o The Relational Model
- o What is PostgreSQL?
- o PostgreSQL – Data Types
- o Arrays Functions and Operators
- **Understanding Basic PostgreSQL Syntax**
 - o The Relational Model
 - o Basic SQL Commands - SELECT
 - o Basic SQL Commands - INSERT
 - o Basic SQL Commands - UPDATE
 - o Basic SQL Commands – DELETE

□

Querying Data with the SELECT Statement

- o Wildcards (% , _)
- o The SELECT List
- o SELECT List Wildcard (*)
- o The FROM Clause
- o How to Constrain the Result Set
- o DISTINCT and NOT DISTINCT

Arrays Functions and Operators

- o array_append
- o array_cat
- o array_lower
- o array_to_string
- o array_agg
- o every, Count, sum, avg
- o Array Operators

Filtering Results with the Where Clause

- o WHERE Clause
- o Boolean Operators
- o The AND Keyword
- o The OR Keyword
- o Other Boolean Operators BETWEEN, LIKE, IN, IS, IS NOT

□

Shaping Results with ORDER BY and GROUP BY

- o ORDER BY
- o Set Functions
- o Set Function And Qualifiers
- o GROUP BY
- o HAVING clause

□

Matching Different Data Tables with JOINS

- o Table Aliases
- o CROSS JOIN
- o INNER JOIN
- o OUTER JOINs
- o LEFT OUTER JOIN
- o RIGHT OUTER JOIN
- o FULL OUTER JOIN
- o SELF JOIN
- o Natural Join

□

Creating Database Tables

- o CREATE DATABASE
- o CREATE TABLE
- o NULL Values
- o PRIMARY KEY
- o CONSTRAINT
- o ALTER TABLE
- o DROP TABLE

□

PostgreSQL Transactions

- o BEGIN, COMMIT, ROLLBACK

□

PostgreSQL Constraints

- o CHECK, UNIQUE, NOT NULL

- Introduction to JDBC
 - o Connection, Statement, PreparedStatement, ResultSet

JPA with Hibernate 3.0

Program Duration: 2 days

Content:

- **Introduction**
 - Introduction & overview of data persistence
 - Overview of ORM tools
 - Understanding JPA
 - JPA Specifications
- **Entities**
 - Requirements for Entity Classes
 - Persistent Fields and Properties in Entity Classes
 - Persistent Fields
 - Persistent Properties
 - Using Collections in Entity Fields and Properties
 - Validating Persistent Fields and Properties
 - Primary Keys in Entities
- **Managing Entities**
 - The EntityManager Interface
 - Container-Managed Entity Managers
 - Application-Managed Entity Managers
 - Finding Entities Using the EntityManager
 - Managing an Entity Instance's Lifecycle
 - Persisting Entity Instances
 - Removing Entity Instances
 - Synchronizing Entity Data to the Database
 - Persistence Units
- **Querying Entities**
 - Java Persistence query language (JPQL)
 - Criteria API
- **Entity Relationships**
 - Direction in Entity Relationships
 - Bidirectional Relationships
 - Unidirectional Relationships
 - Queries and Relationship Direction
 - Cascade Operations and Relationships

Spring 5.0

Program Duration: 6 days

Contents:

1. Spring Core

Spring Core Introduction / Overview

- Shortcomings of Java EE and the Need for Loose Coupling
- Managing Beans, The Spring Container, Inversion of Control
- The Factory Pattern
- Configuration Metadata - XML, @Component, Auto-Detecting Beans
- Dependencies and Dependency Injection (DI) with the BeanFactory
- Setter Injection

Spring Container

- The Spring Managed Bean Lifecycle
- Autowiring Dependencies

Dependency Injection

- Using the Application Context
- Constructor Injection
- Factory Methods
- Crucial Namespaces 'p' and 'c'
- Configuring Collections

Metadata / Configuration

- Annotation Configuration @Autowired, @Required, @Resource
- @Component, Component Scans. Component Filters
- Life Cycle Annotations
- Java Configuration, @Configuration, XML free configuration
- The Annotation Config Application Context

2. Spring Boot

SPRING BOOT Introduction

- Spring Boot starters, CLI, Gradle plugin
- Application class
- @SpringBootApplication
- Dependency injection, component scans, Configuration
- Externalize your configuration using application.properties
- Context Root and Management ports
- Logging

Using Spring Boot

- Build Systems, Structuring Your Code, Configuration, Spring Beans and Dependency Injection, and more.

Spring Boot Essentials

- Application Development, Configuration, Embedded Servers, Data Access, and many more
- Common application properties
- Auto-configuration classes
- Spring Boot Dependencies

3. Spring Data JPA

- Spring Data JPA Intro & Overview
- Core Concepts, @Repository, @Resource
- Defining Query methods
- Query Creation
- Using JPA Named Queries
- Defining Repository Interfaces
- Creating Repository instances
- JPA Repositories
- Persisting Entities
- Transactions

4. Spring Data REST

- Introduction & Overview
- Adding Spring Data REST to a Spring Boot Project
- Configuring Spring Data REST
- Repository resources, Default Status Codes, Http methods
- Spring Data REST Associations
- Define Query methods

5. Introduction to Spring Security with Demo

6. Introduction to Spring Microservices with Demo

HTML 5, CSS 3 with Bootstrap, Javascript, TypeScript

Program Duration: 4 days

Contents:

HTML 5:

- HTML Basics
 - Understand the structure of an HTML page.
 - New Semantic Elements in HTML 5
 - Learn to apply physical/logical character effects.
 - Learn to manage document spacing.
- Tables
 - Understand the structure of an HTML table.
 - Learn to control table format like cell spanning, cell spacing, border
- List
 - Numbered List
 - Bulleted List
- Working with Links
 - Understand the working of hyperlinks in web pages.
 - Learn to create hyperlinks in web pages.
 - Add hyperlinks to list items and table contents.
- Image Handling
 - Understand the role of images in web pages
 - Learn to add images to web pages
 - Learn to use images as hyperlinks
- Frames
 - Understand the need for frames in web pages.
 - Learn to create and work with frames.
- HTML Forms for User Input
 - Understand the role of forms in web pages
 - Understand various HTML elements used in forms.
 - Single line text field
 - Text area
 - Check box
 - Radio buttons
 - Password fields
 - Pull-down menus
 - File selector dialog box
- New Form Elements
 - Understand the new HTML form elements such as date, number, range, email, search and datalist
 - Understand audio, video, article tags

CSS 3

- **Introduction to Cascading Style Sheets 3.0**
 - What CSS can do
 - CSS Syntax
 - Types of CSS
- **Working with Text and Fonts**
 - Text Formatting
 - Text Effects
 - Fonts
- **CSS Selectors**
 - Type Selector
 - Universal Selector
 - ID Selector
- o Class selector
- **Colors and Borders**
 - Background
 - Multiple Background
 - Colors RGB and RGBA
 - HSL and HSLA
 - Borders
 - Rounded Corners
 - Applying Shadows in border
- Implementing CSS3 in the "Real World"
 - o Modernizr
 - o HTML5 Shims
 - o SASS, and Other CSS Preprocessors
 - o CSS Grid Systems
 - o CSS Frameworks

BootStrap

- **Introduction to Bootstrap**
 - Introduction
 - Getting Started with Bootstrap
- **Bootstrap Basics**
 - Bootstrap grid system
 - Bootstrap Basic Components
- **Bootstrap Components**
 - Page Header
 - Breadcrumb
 - Button Groups
 - Dropdown
 - Nav & Navbars
- **JavaScript Essentials**
- **ES6 & Typescript**

- Var, Let and Const keyword
- Arrow functions, default arguments
- Template Strings, String methods
- Object de-structuring
- Spread and Rest operator
- Typescript Fundamentals
- Types & type assertions, Creating custom object types, function types
- Typescript OOPS - Classes, Interfaces, Constructor, etc

Angular 7

Duration : 6 days

Contents:

- **Introduction to Angular Framework**
 - Introduction to Angular Framework, History & Overview
 - Environment Setup, Angular CLI, Installing Angular CLI
 - NPM commands & package.json
 - Bootstrapping Angular App, Components, AppModule
 - Project Setup, Editor Environments
 - First Angular App & Directory Structure
 - Angular Fundamentals, Building Blocks
 - MetaData
- **Essentials of Angular**
 - Component Basics
 - Setting up the templates
 - Creating Components using CLI
 - Nesting Components
 - Data Binding - Property & Event Binding, String Interpolation, Style binding
 - Two-way data binding
 - Input Properties, Output Properties, Passing Event Data
- **Templates, Styles & Directives**
 - Template, Styles, View Encapsulation, adding bootstrap to angular app
 - Built-in Directives, Creating Attribute Directive
 - Using Renderer to build attribute directive
 - Host Listener to listen to Host Events
 - Using Host Binding to bind to Host Properties
- **Pipes, Services & Dependency Injection**
 - In-built Pipes, Creating a Custom Pipes

- Services & Dependency Injections
- Creating Data Service
- Understanding Hierarchical Injector

- **Template-Driven and Reactive Forms**
 - Template-Driven vs Reactive Approach
 - Understanding Form State
 - Built-in Validators & Using HTML5 Validation
 - Grouping Form Controls
 - FormGroup, FormControl, FormBuilder
 - Forms with Reactive Approach
 - Predefined Validators & Custom Validators
 - Showing validation errors

- **Components Deep Dive / Routing**
 - Component Life Cycle Hooks
 - Reusable components in angular using <ng-content>
 - Navigating with Router links
 - Understanding Navigation Paths
 - Navigating Programmatically
 - Passing Parameters to Routes
 - Passing Query Parameters and Fragments
 - Setting up Child (Nested) Routes
 - Outsourcing Route Configuration (create custom module)

- **Http Requests / Observables**
 - HTTP Requests
 - Sending GET Requests
 - Sending a PUT Request
 - Using the Returned Data
 - Catching Http Errors
 - Basics of Observables & Promises



Ms. Bhavini Shah
Course Co-ordinator