# Java 8 Course contents

**By Dr. Vishwanath Rao**

## 1. Java Platform Overview

- Defining how the Java language achieves platform independence
- Differentiating between the Java ME, Java SE, and Java EE Platforms
- Evaluating Java libraries, middle-ware, and database options
- Defining how the Java language continues to evolve

## 2. Java Syntax and Class Review

- Creating simple Java classes
- Creating primitive variables
- Using operators
- Creating and manipulate strings
- Using if-else and switch statements
- Iterating with loops: while,do-while,for,enhanced for
- Creating arrays
- Using Java fields, constructors, and methods

## 3. Encapsulation and Subclassing

- Using encapsulation in Java class design
- Modeling business problems using Java classes
- Making classes immutable
- Creating and use Java subclasses
- Overloading methods

## 4. Overriding Methods, Polymorphism, and Static Classes

- Using access levels: private, protected, default, and public
- Overriding methods
- Using virtual method invocation
- Using varargs to specify variable arguments
- Using the instanceof operator to compare object types
- Using upward and downward casts
- Modeling business problems by using the static keyword
- Implementing the singleton design pattern

## 5. Abstract and Nested Classes

- Designing general-purpose base classes by using abstract classes
- Constructing abstract Java classes and subclasses
- Applying final keyword in Java
- Distinguish between top-level and nested classes

## 6. Interfaces and Lambda Expressions

- Defining a Java interface

- Choosing between interface inheritance and class inheritance
- Extending an interface
- Defaulting methods
- Anonymous inner classes
- Defining a Lambda Expression

### 7. Collections and Generics

- Creating a custom generic class
- Using the type inference diamond to create an object
- Creating a collection by using generics
- Implementing an ArrayList
- Implementing a TreeSet
- Implementing a HashMap
- Implementing a Deque
- Ordering collections

### 8. Collections Streams, and Filters

- Describing the Builder pattern
- Iterating through a collection using lambda syntax
- Describing the Stream interface
- Filtering a collection using lambda expressions
- Calling an existing method using a method reference
- Chaining multiple methods together
- Defining pipelines in terms of lambdas and collections

### 9. Lambda Built-in Functional Interfaces

- Listing the built-in interfaces included in java.util.function
- Core interfaces - Predicate, Consumer, Function, Supplier
- Using primitive versions of base interfaces
- Using binary versions of base interfaces

### 10. Lambda Operations

- Extracting data from an object using map
- Describing the types of stream operations
- Describing the Optional class
- Describing lazy processing
- Sorting a stream
- Saving results to a collection using the collect method
- Grouping and partition data using the Collectors class

### 11. Exceptions and Assertions

- Defining the purpose of Java exceptions
- Using the try and throw statements
- Using the catch, multi-catch, and finally clauses
- Autoclose resources with a try-with-resources statement
- Recognizing common exception classes and categories
- Creating custom exceptions

- Testing invariants by using assertions

## 12. Java Date/Time API

- Creating and manage date-based events
- Creating and manage time-based events
- Combining date and time into a single object
- Working with dates and times across time zones
- Managing changes resulting from daylight savings
- Defining and create timestamps, periods and durations
- Applying formatting to local and zoned dates and times

## 13. I/O Fundamentals

- Describing the basics of input and output in Java
- Read and write data from the console
- Using streams to read and write files
- Writing and read objects using serialization

## 14. File I/O (NIO.2)

- Using the Path interface to operate on file and directory paths
- Using the Files class to check, delete, copy, or move a file or directory
- Using Stream API with NIO2

## 15. Concurrency

- Describing operating system task scheduling
- Creating worker threads using Runnable and Callable
- Using an ExecutorService to concurrently execute tasks
- Identifying potential threading problems
- Using synchronized and concurrent atomic to manage atomicity
- Using monitor locks to control the order of thread execution
- Using the java.util.concurrent collections

## 16. The Fork-Join Framework

- Parallelism
- The need for Fork-Join
- Work stealing
- RecursiveTask
- RecursiveTask

## 17. Parallel Streams

- Reviewing the key characteristics of streams
- Describing how to make a stream pipeline execute in parallel
- List the key assumptions needed to use a parallel pipeline
- Defining reduction
- Describing why reduction requires an associative function
- Calculating a value using reduce
- Describing the process for decomposing and then merging work
- Listing the key performance considerations for parallel streams

## 18. Database Applications with JDBC

- Defining the layout of the JDBC API
- Connecting to a database by using a JDBC driver
- Submitting queries and get results from the database
- Specifying JDBC driver information externally
- Performing CRUD operations using the JDBC API

## 19. Localization

- Describing the advantages of localizing an application
- Defining what a locale represents
- Read and set the locale by using the Locale object
- Building a resource bundle for each locale
- Calling a resource bundle from an application
- Changing the locale for a resource bundle