
venolearn
EXPERIENCE BEYOND IT TRAINING

What is JAVA/J2EE?

JAVA (Java Standard Edition) is a collection of Java APIs used to run desktop applications whereas J2EE (Java Enterprise Edition) is used for applications that run on servers, such as websites.

The course will focus on the practical application JAVA/J2EE. It will also include JDBC (Java Database Connectivity) which facilitates connectivity between Java and databases. Also, JSPs (Java Server Pages) that allow Java to work in combination with HTML will be a part of this training module.

Why JAVA/J2EE?

Almost 90% of the Fortune 500 companies use Java for their desktop as well as web applications, which clearly defines a large scope for the programming language. Android apps are also developed using Java. It is a highly portable language finding its use across various industries such as gaming, trading applications, embedded and electronic systems, mobile apps, TV devices and more.

It is the number one choice of Developers. The language is machine-independent. You write once and run it anywhere!

What are the benefits of learning JAVA/J2EE?

- Java is a beginner-friendly language and simple to use.
- Java is the most in-demand language and will continue to remain so for a long time.
- Java is fast and offers optimized performance with its Just-in-time compilers and improved JVMs.
- Learning JAVA/J2EE opens up a plethora of career opportunities for you across different industries. When it comes to finding jobs as a Java Developer, no technology comes as close.

Course Duration:

- 45 Working days

Course Syllabus:

Object Oriented Programming (OOPS) concepts

- OOPS concepts and terminology
- Advantage of OOPS
- Fundamentals of OOPS

Core Java Programming Introduction of Java

- What is Java?
- How to Get Java
- A First Java Program
- Compiling and Interpreting Applications
- The JDK Directory Structure
- Using Eclipse

Data types and Variables

- Primitive Datatypes, Declarations,Variable Names
- Numeric Literals, Character Literals
- String formatting and Parsing
- String Literals
- Arrays, Non-Primitive Datatypes
- The Dot Operator

Methods

- Methods
- Calling Methods
- Defining Methods
- Method Parameters Scope
- So, Why All the static?

Operators and Expressions

- Expressions
- Assignment Operator

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Increment and Decrement Operators
- Operate-Assign Operators (+=, etc.)
- The Conditional Operator
- Operator Precedence
- The Cast Operator

Object-Oriented Programming

- Introduction to Object-Oriented Programming
- Classes and Objects
- Fields and Methods
- Encapsulation
- Access Control

Control Flow Statements

- Statements
- Conditional (if) Statements
- Adding an else if Conditional (switch) Statements
- while and do-while Loops
- for Loops
- A for Loop Diagram
- Enhanced for Loop
- The break Statement
- The continue Statement

Objects and Classes

- Defining a Class
- Creating an Object
- Instance Data and Class Data
- Methods
- Constructors
- Access Modifiers

Using Java Objects

- StringBuilder and StringBuffer
- Methods and Messages
- Parameter Passing
- Comparing and Identifying Objects

Inheritance in Java

- Inheritance in Java
- Methods of Overriding
- Polymorphism
- Super keyword
- The Object Class

Java Files and I/O

- Reading and Writing to Files
- Input and Output Stream
- File handling Classes

Interfaces and Abstract Classes

- Separating Interface and Implementation
- Defining Interfaces
- Implementing and Extending Interfaces
- Abstract Classes

Packages

- The import Statement
- Static Imports
- CLASSPATH and Import
- Defining Packages
- Package Scope

Lambda Built-in Functional Interfaces

- java.util.function package
- Use primitive versions of functional Interface
- Use binary versions of functional Interface
- Use the UnaryOperator Interface

Exception Handling

- Exceptions Overview
- Catching Exceptions
- The Finally Block
- Exception Methods
- Declaring Exceptions
- Defining and Throwing Exceptions
- Errors and Runtime Exceptions
- Assertions

Collection Framework

- The Collection Framework
- The Set Interface
- Set Implementation Classes
- The List Interface
- List Implementation Classes
- The Map Interface
- Map Implementation Classes
- Utility Classes
- Generics
- Primitive wrapper Classes

Inner Classes

- Inner Classes
- Member Classes
- Local Classes

Introducing to Threads

- Non-Threaded Applications
- Threaded Applications
- Thread Lifecycle
- Phases of Thread Lifecycle
- Coordinating Threads

Swing

- Swing GUI Components
- Using Swing API

Applet

- Life Cycle of an Applet
- A "Hello World" Applet

JDBC

- Getting Information from Database
- Obtaining Result Set Information
- Connecting a Java program to a Database

Agile Scrum overview

- Introduction to Agile Methodology
- Scrum & its characteristics
- Sprints in Scrum
- Overview of Scrum Artifacts & Ceremonies

ADVANCED JAVA

Java MVC Architecture

- The Model
- The View pt1

- The View pt2
- The Controller

Servlets

- What is a web application?
- Java Servlets
- What is a Servlet?
- Servlet Lifecycle
- Servlet Context
- Session management
- Building the first Servlet
- Deploying the Servlet
- Examples – Servlets

JSP

- What is a JSP Page?
- Basic HTML Tags
- JSP Tag library
- JSP Page Life-cycle
- Creating the first Dynamic web page using JSP

SQL

- Introduction of SQL
- SQL - RDBMS Concepts
- SQL DataType
- SQL Operators
- SQL Statements
- SQL Functions

Struts2

- Introductions to the MVC1 & MVC2 Architecture Struts2
- Overview of Struts Framework
- Components of Model, View and Controller in Struts Framework

- Action Classes
- Handling Application Requests
- Deployment Descriptors

Spring

- Introduction of Spring Framework
- Spring Framework Architecture
- Spring bean Wiring
- AOP with Spring
- Transactions management
- Spring with database

Hibernate

- Introductions to Hibernate
- Object Related Mapping
- Persistent Classes
- Mapping Collections
- Hibernate Query Language
- Caching and Transactions
- Hibernate with Web Applications