

6 MONTHS

INTRODUCTION TO JAVA

- • • Why Java was Developed
- • • Application Areas of Java
- • • History of Java
- • • Platform Independency in Java
- • • USP of Java: Java Features
- • • Sun-Oracle Deal
- • • Different Java Platforms
- • • Difference between JDK,JRE,JVM
- • • Java Versions
- • • JVM Architecture
- • • Installing Java on Windows
- • • Understanding Path Variable: Why Set Path

CREATING FIRST JAVA PROGRAM

- • • Understanding Text Editors to Write Programs
- • • How to compile java file
- • • Byte Code and class file
- • • How to run class file

JAVA LANGUAGE FUNDAMENTALS

- • • Identifiers
- • • Keywords
- • • Variables
- • • Literals
- • • Data Types
- • • Operators
- • • Comments
- • • Looping Statements
- • • Condition Statements
- • • Type Casting

OOP IMPLEMENTATION (PIE)

- • • Why OOP
- • • OOP Concepts with Real life examples
- • • Class& it's Syntax
- • • Object& it's Syntax

- • • Reference Variable
- • • Constructors
- • • Instance(Non-Static)& Static Variables
- • • Instance(Non-Static) & Static Methods
- • • this Keyword and it's usages
- • • Object & Static Initializers(Anonymous Blocks)
- • • Understanding '+' Operator

EXPERT

6 MONTHS

- • • Inheritance& it's Syntax
- • • Types of Inheritance
- • • Object Class as Root of Java Class Hierarchy
- • • Variable Hiding
- • • Method Hiding
- • • Method Overriding
- • • Method Overloading
- • • Super keyword and it's usages
- • • Final keyword and it's usages
- • • Constructor Chaining
- • • Upcasting and Downcasting
- • • Static &Dynamic Binding
- • • Run Time Polymorphism
- • • Abstract Keyword(Abstract classes and methods)
- • • Understanding Interfaces
- • • Implementation of Encapsulation
- • • Association with Implementation

PACKAGES

- • • Understanding Packages
- • • Setting Class path
- • • Reading Input from Keyboard
- • • Access Modifiers

NESTED TYPES

- • • Static Nested Class
- • • Non-static Nested Class
- • • Local Class
- • • Anonymous Class

- • • Nested Interface

ARRAYS

- • • General Definition of Array
- • • Advantages from Array
- • • Arrays in Java
- • • 1-d Arrays
- • • 2-d Arrays
- • • Jagged Arrays
- • • Array of reference type
- • • Operations on Arrays

COMMAND LINE ARGUMENTS AND WRAPPER CLASSES

- • • How to read command line arguments
- • • Wrapper Classes
- • • Parsing of Numeric Strings
- • • String representation of Primitives

EXCEPTION HANDLING

- • • Types of Runtime Errors
- • • Understanding Exceptions
- • • Exception Class Hierarchy
- • • Try & Catch Blocks
- • • Patterns of Catch Block
- • • Nested Try statements
- • • Throw, throws and finally
- • • Creating Custom Exceptions
- • • Checked & Unchecked Exceptions
- • • Assertion

WORKING WITH STRINGS

- • • What is String
- • • String Class
- • • Creating String Object
- • • Operations on String
- • • String Buffer Class and it's Methods
- • • Difference between String and StringBuffer class
- • • String Builder Class and it's Methods
- • • Difference between StringBuffer and StringBuilder

SWING

- • • Introduction to AWT
- • • Introduction to Swing Components
- • • Look And Feel of Swing Components
- • • MVC Architecture of Swing Components

- • • Working with Image
- • • Advance Swing Components
- • • JOptionPane, JTree, JTable, JTabbedPane
- • • JFileChooser, JcolorChooser
- • • Menu Components
- • • JMenu
- • • JMenuItem
- • • JMenuBar

MULTITHREADED PROGRAMMING

- • • Multitasking: Why Concurrent Execution
- • • Multiprocessing v/s Multithreading
- • • Main Thread (Default Java Thread)
- • • Creating Child Threads and understanding context switching
- • • Thread States
- • • Thread Group
- • • Thread Synchronization: Methods and Blocks
- • • Inter-Thread communication
- • • Daemon Threads
- • • Deadlock

I/O STREAMS

- • • What is I/O
- • • Why Need Streams
- • • Byte Streams and Character Streams
- • • Read/Write operations with file
- • • Scanner Class
- • • Object Serialization & Deserialization
- • • Transient keyword
- • • File Class and it's Methods

SOCKET PROGRAMMING

- • • Understanding Fundamentals of a Network
- • • Socket and ServerSocket Classes
- • • InetAddress Class
- • • DatagramSocket and DatagramPacket Classes
- • • URL, URLConnection, HttpURLConnection Classes

REFLECTION

- • • Understanding the Need Of Reflection
- • • Getting information about class's modifiers, fields, methods, constructors and super classes
- • • Finding out constant and method declaration belong to an interface
- • • Creating an instance of the class whose name is not known until runtime

- • • Getting and setting values of an object's field if field name is unknown until runtime
- • • Invoking a method on an object if the method is unknown until runtime
- • • Invoking Private Methods

EXTENDED & UTILITY CONCEPTS

- • • Generics
- • • Lambda Expression
- • • Annotations
- • • Object Cloning
- • • Vargs
- • • Static-import
- • • Enum
- • • Static, Default and Private Methods of Interface
- • • Var Type
- • • Java Modules

COLLECTIONS FRAMEWORK

- • • What is Collection?
- • • What is Framework?
- • • Collections Framework
- • • Core Interfaces
- • • Collection, List, Queue, Deque
- • • Set, NavigableSet, SortedSet
- • • Map, NavigableMap, SortedMap
- • • Core Classes
- • • ArrayList, LinkedList, PriorityQueue, ArrayDeque
- • • HashSet, LinkedHashSet, TreeSet,
- • • HashMap, IdentityHashMap, WeakHashMap, LinkedHashMap, Tree Map
- • • Accessing a Collection via an Iterator
- • • Accessing List via ListIterator
- • • Accessing a Collection via for each loop
- • • Working with User Defined Objects
- • • The Comparator and Comparable Interfaces
- • • The Legacy classes and Interfaces.
- • • Enumeration, Vector, Stack
- • • Hashtable, Properties

DATE & TIME API

- • • java.util.Date
- • • java.util.Calendar
- • • java.sql.Date

JODA API

- • • java.time.LocalDate
- • • java.time.LocalDateTime
- • • java.time.LocalTime

SYSTEM PROPERTIES & INTERNATIONALIZATION (I18N)

- • • Understanding Locale
- • • Resource Bundle
- • • Usage of properties file
- • • Fetching text from Resource Bundle
- • • Displaying the text in HINDI
- • • Displaying date in Hindi

INTRODUCTION TO SQL (PROJECT BASED) DATABASE PROGRAMMING USING JDBC

- • • Need Of JDBC
- • • JDBC Drivers
- • • Statement, PreparedStatement, CallableStatement
- • • Scrollable and Updatable ResultSet
- • • Batch Updates
- • • Transaction
- • • Metadata

JAVA EE(JAVA PLATFORM ENTERPRISE EDITION)

- • • Understanding the Concept of Java EE : JEE Specification
- • • Java EE Architecture
- • • Single Tier
- • • Two Tier
- • • Three Tier
- • • N-Tier
- • • Java EE Components
- • • Web Components
- • • Distributed(Business) Components
- • • Java EE Containers& Servers
- • • Web Container& Web Server(Apache Tomcat)
- • • EJB Container& Application Server(Weblogic,Glassfish,Websphere)
- • • Java EE Services
- • • JNDI Service
- • • Java Transaction Service
- • • JAAS
- • • JMS

JAVA SERVLET

- • • Introduction to web programming
- • • Role of Servlet in web programming

- • • Servlet Lifecycle
- • • Servlet with Annotations
- • • @WebServlet
- • • @WebInitParam
- • • @WebListener
- • • @WebFilter
- • • @MultipartConfig
- • • Request Dispatching
- • • Parameters & Attributes and their differences
- • • ServletConfig and ServletContext
- • • File Uploading and Downloading
- • • Session Tracking&State Management
- • • Cookie
- • • Url Rewriting
- • • Hidden Form Field
- • • Session Object
- • • Events & Listeners
- • • Dependency Injection
- • • Refreshing Servlet
- • • Filters

JAVA SERVER PAGES (JSP) & JSTL

- • • JSP Architecture
- • • JSP Elements
- • • JSP Objects
- • • Understanding JavaBeans
- • • Custom Tags
- • • Using tags of JSTL
- • • Expression Language

PROJECT CLASSES

- • • Front End Coding
- • • FORM DESIGNING
- • • HTML
- • • CSS
- • • JAVA SCRIPT
- • • BOOTSTRAP
- • • Back End Coding
- • • DATABASE DESIGNING
- • • Connecting forms to database
- • • Writing Business Logic
- • • Project Hosting

DESIGN PATTERN

- • • Why Design Patterns...?
- • • Front Controller
- • • Composite View
- • • Factory Pattern
- • • Singleton Pattern
- • • DAO Pattern

JAVA MAIL API

- • • Email System and Protocols
- • • Sending & Receiving Mails
- • • Handling Attachments

INTRODUCTION TO DISTRIBUTED PROGRAMMING

- • • RMI
- • • Web Services

INTRODUCTION TO RESTFULL SERVICES

- • • @PathParam
- • • @Path
- • • @FormParam
- • • @QueryParam
- • • @DefaultValue

OVERVIEW OF JPA FRAMEWORK

SPRING

- • • What is Spring?
- • • Spring modules
- • • Understanding dependency Injection
- • • Applying aspect-oriented programming

BASIC BEAN WIRING

- • • Containing your Bean
- • • Creating bean
- • • Injecting into bean properties
- • • Auto wiring
- • • Controlling bean creation

ADVANCED BEAN WIRING

- • • Declaring parent and Child Bean
- • • Applying method injection
- • • Injecting Non-spring Beans
- • • Registering Custom property editors

ADVISING BEANS

- • • Introducing AOP

- • • Creating classic spring aspects
- • • Creating advice
- • • Defining Pointcuts and Advisors
- • • Using proxyFactory Bean
- • • Autoproxying

HITTING THE DATABASE

- • • Learning spring's data Access Philosophy
- • • Configuring a data source
- • • Using JDBC with Spring
- • • Working with JDBC Templates
- • • Using Spring's DAO Support Classes for JDBC
- • • Integrating Hibernate with Spring
- • • Caching

INTRODUCTION TO MVC

- • • Define MVC
- • • Hibernate Injection
- • • Spring Annotation
- • • Spring Controller

MAVEN DEPLOYMENT

- • • Maven Configuration
- • • Converting Maven to Eclipse
- • • Various Maven Command

SPRING REST API

- • • Creating Rest
- • • Consuming Rest
- • • Calling on Client

BUILDING CONTRACT-FIRST WEB SERVICES IN SPRING

- • • Introducing Spring-WS
- • • Defining Contract (First!)
- • • Handling messages with service endpoints
- • • Wiring it all together
- • • Consuming Spring-WS Web services

SPRING OBJECT/XML MAPPER

SPRING BOOT

- • • Project Creation
- • • Boot Elements
- • • Boot Services
- • • Boot Annotation

INTRODUCTION TO ORM

- • • Need of ORM
- • • Problems using JDBC Directly
- • • ORM Implementation

INTRODUCTION TO HIBERNATE

- • • Hibernate Architecture
- • • Hibernate configuration
- • • Hibernate's Support for Other Technologies
- • • Installing Hibernate
- • • A “Hello world” stand alone application
- • • A Servlet–Based Hibernate application

CREATING PERSISTING CLASSES

- • • Mapping a basic Java Class
- • • Mapping a Class with Binary Data
- • • Mapping a Serializable Class
- • • Mapping a class with Data/ calendar attributes
- • • Mapping a Read-only class
- • • Mapping a class using Versioning /Timestamps

MAPPING INHERITENCE WITH JAVA CLASSES

- • • Table-Per –class Hierarchy Mapping
- • • Table-Per –subclass Hierarchy Mapping
- • • Table-Per –concrete-subclass Hierarchy Mapping
- • • Persistence interfaces

WORKING WITH COLLECTIONS

- • • Associations
- • • Lazy initialization
- • • Mapping Maps/Sorted Maps
- • • Mapping Sets/Sorted Sets
- • • Mapping lists
- • • Mapping Arrays
- • • Mapping a Bidirectional Association

SCALAR QUERIES AND HIBERNATE QUERY LANGUAGE

- • • Queries
- • • Named Queries
- • • SQL Queries
- • • Hibernate Queries language