Core Java

Core Java Training Overview

This Core Java Training is by the Real-Time Professionals and Teaching Experts.

- Entire SCJP syllabus will be covered
- Every program execution will be explained with Compiler and JVM Architectures
- Every program memory diagram will be clearly explained with JVM Architecture
- 1000+ Programs will be covered in training as well as in practice material
- Entire list of interview questions will be covered on every concept
- Every concept will be clearly explained with real-time project scenarios
- Every concept will be explained with MVC and LC-RP Architectures
- Therefore you will get good knowledge in designing and developing projects
- So that you can clear all interviews as a fresher or as an experienced

Core Java Training Prerequisites

- No Prerequisite, Anyone Can Learn

Core Java Course Duration

- Normal Track 60 Working days, daily one and half hour
- Fast Track 40 Working days, daily two hours
- Core Java Training Course Overview

Core Java Training Content Overview

Java Language, OOPS, Programming

1. Introduction to Java and OOPS
2. Java Tokens- Comments, Identifiers, Keywords, Separators
3. Working with Java Editor Softwares – Editplus, NetBeans, Eclipse
4. Packages with static imports
5. Working with jar
6. Modifiers – File level, Access level and Non-access level

7. Datatypes, Literals, Variables, Type Conversion, Casting & Promotion
8. Reading runtime values from keyboard and Properties File
9. Operators and Control Statements
10. Method and Types of methods
11. Variable and Types of Variables
12. Constructor and Types of constructors
13. Block and Types of Blocks
14. Declarations, Invocations and Executions
15. Compiler & JVM Architecture with Reflection API
16. Static Members and their execution control flow
17. Non-Static Members and their execution control flow
18. Final Variables and their rules
19. Classes and Types of classes
20. OOPS- Fundamentals, Models, Relations and Principles
21. Coupling and Cohesion (MVC and LCRP Architectures)
22. Types of objects & Garbage Collection
23. Arrays and Var-arg types
24. Enum and Annotation
25. Design Patterns

Java API and Project

1. API and API Documentation
2. Fundamental Classes – Object, Class, System, Runtime
3. String Handling
4. Exception Handling and Assertions
5. Multithreading with JVM Architecture
6. IO Streams (File IO)

7. Networking (Socket Programming)
8. Wrapper Classes with Auto boxing and unboxing
9. Collections with Generics
10. Java 5, 6, 7, 8 new features
11. Inner classes
12. AWT, Swings, Applet
13. Regular Expressions
14. Formatting date, time (java.text package)

Developing Project using Core Java.

Advanced Java

About Advanced Java Training

The course builds a strong understanding of JDBC Technology. It gives in to demonstrate why Servlets are the cornerstone of Java's Web platform. It then shows how JSP is built on the Servlet architecture. Additionally, the class shows students how to use JSTL, custom tags and expression language to reduce Java code in Web pages while adding tremendous power and capability to those pages. The class culminates in an exploration of Java MVC frameworks like Struts at a high level.

This is not a class that focuses on theory. Participants will find the course is loaded with practical labs and simulations. After taking this class, developers will be able to build Web applications that perform well, are scalable, and that are easier to maintain.

Advanced Java Training Course Prerequisite

Basic Knowledge of Core Java is required. An understanding of Web technologies like HTML and HTTP is helpful.

Advanced Java Training Course Objective

Learn the fundamentals of JDBC and using the different interfaces in the JDBC API. Learn how to use Java servlets in the role of Web application control. Identify the options to state management in a Java Web application and understand the pros/cons of each. Understand how JSPs can help to separate Web logic and functionality from page layout. Explore how to make JSPs smaller and more powerful with JSTL, custom tags and expression language. Explore strategies in the exchange of data between Web pages (views) and business processing (model). Learn the meaning and importance of MVC

Advanced Java Training Course Duration

60 Working days, daily one and half hour

Advanced Java Training Course Content Overview

Introduction

- About Java Platforms
- Applications of Java
- About API (Application Programming Interface)
- Static Blocks
- Introduction to interfaces
- Runtime Polymorphism

Persistence

- What is Persistence?
- File management system
- Disadvantages of File management system
- Database management system
- Advantages of Database management system

JDBC

- Introduction to JDBC
- Why JDBC? & ODBC Vs JDBC
- Approch-1 Vendor specific library
- Disadvantages
- Approch-2 X/Open standards
- Disadvantages
- Approch-3 ODBC standards
- Disadvantages
- Approch-4 JDBC standards
- Advantages

JDBC API and JDBC Drivers

- About JDBC API
- What is a JDBC Driver?
- JDBC architecture
- Types of JDBC Drivers
- JDBC-ODBC bridge driver
- Advantages & Disadvantages
- Native API Partly Java Driver
- Advantages & Disadvantages
- Net-Protocol all/pure java driver
- Advantages & Disadvantages
- Native-Protocol pure java driver
- Advantages

JDBC API

- JDBC Packages
- sql, Javax.sql
- Interfaces and classes in above packages
- Versions of JDBC API
- Steps for developing JDBC application

SQL

- Introduction to SQL
- Types of SQL queries

Establishing Connection to Database

- About DriverManager class
- Methods of DriverManager
- What is registering driver with DriverManager

- Different methods of registering driver with DriverManager
- About getConnection method of DriverManager
- Syntax of URL to communicate with database
- Autoloading of class driver in JDBC 4.0
- Example with Application

Statement interface

- What is Statement? & Need of Statement
- Characteristics of Statement
- How to get Statement object?
- Methods of Statement
- Passing SQL statements to Database
- About execute, executeUpdate, executeQuery, largeExecute Updates methods
- Example with Application

PreparedStatement interface

- What is PreparedStatement?
- Need of PreparedStatement
- Difference between Statement and PreparedStatement
- How to get PreparedStaement object
- Defining parameters
- How to pass values to PerparedStatement object
- What is SQL Injection attack & how to overcome
- Example with Application

Resultset Interface

- What is ResultSet?,
- Need of ResultSet
- How to get resultset object

- Methods of ResultSet
- Reading data, Various types of ResultSet
- Resultset types
- TYPE_SCROLL_SENSITIVE
- TYPE_SCROLL_INSENSITIVE
- TYPE_FORWARD_ONLY
- Resultset modes
- CONCUR_READ_ONLY, CONCUR_UPDATABLE
- Example with Application

SQL 99 Datatypes

- BLOB
- Inserting blob type, Reading blob type
- CLOB
- Inserting clob type, Reading clob type
- Array, Inserting array type
- Reading array type
- Object, Inserting object
- Reading object
- Example with Application

Metadata

- Resultset Metadata
- Need of ResultsetMetaData
- How to get metadata
- DatabaseMetadata
- How to get database metadata
- Parameterizedmetadata

- How to get parameterizedmetadata
- Example with Application

CallableStatement

- About CallableStatement
- Advantage of CallableStatement
- Creating CallableStatement object
- Calling Procedure using CallableStatement
- Calling functions using Callable Statement
- About PL/SQL Programming vs Manual on Batch Queries

Batch updates

- What is batch processing
- Batch updates using Statement object
- Batch updates using PreparedStatement object

Transaction Management

- Transaction Definition
- ACID Properties
- Atomicity, Consistency,Isolation, Durability
- Methods in Transaction Management
- setAutoCommit(), setSavePoint()
- commit(), rollback()
- Example with All Application

Connection Pooling

- What is connection pooling?
- Advantages of connection pooling
- Disadvantage of DriverManager
- About DataSource

- JDBC connection pooling
- Isolation Levels
- Example with Application

RowSet

- What is RowSet?
- What is difference between ResultSet and RowSet
- Types of RowSets
- JdbcRowSet
- CachedRowSet
- WebRowSet
- Example with Application

Communicating with difference databases

- Communicating with MYSQL
- Working with CSV files
- Communicating with MS-EXCEL
- Communicating with PostgreSql
- Example with Application

JDBC 4.0 Features

- Auto-loading of JDBC driver class
- Connection management enhancements
- Support for RowIdSQL type
- DataSet implementation of SQL using Annotations
- SQL exception handling enhancements
- SQL XML support
- Example with Application

Working With Properties File

- Working With Date Values
- Procedure To Create Desktop icon (jar file execution)

SERVLET

JEE

- JEE technology
- Components of JEE technology
- What is Enterprises Application
- What is Web Application
- Types of Web Application
- Presentation Oriented Web Application
- Service Oriented Web Application
- About Web Client, About WebServer
- About ApplicationServer
- Diff Between WebServer And Application Server

Introduction

- Server side technologies
- Need of server side technologies
- Client side technologies
- What is Servlet? & Advantages of Servlet
- Difference between CGI and Servlet

Servlet API

- Versions of Servlet API
- Packages of Servlet API
- About Servlet Container
- Responsibilities of Servlet Container

HTTP Protocol

- What is HTTP?
- What is HTTP Request format & Response format

Developing Servlet

- Servlet interface
- Methods of Servlet interface
- Developing servlet by implementing Servlet interface
- Life cycle methods of Servlet
- Webapplication directory structure
- Deployment descriptor file (web.xml)
- What is deployment?
- Types of deployments
- Deployment, Console deployment
- Tool deployment
- Deployment webapplication in tomcat server
- URL patterns

Developing And Deploying Servlet Apps in Diff Servers Like

- Tomcat Server
- JBoss Server / WildFly
- Weblogic Server
- GlassFish

ServletRequest

- Methods of ServletRequest
- About request parameters
- About request headers

ServletResponse

- Methods of ServletResponse

- MIME types
- Generating response

ServletConfig

- What is ServletConfig?
- What is need of ServletConfig
- Methods of ServletConfig
- Different ways of getting ServletConfig object.
- Defining config initial parameters in web.xml

GenericServlet

- What is GenericServlet?
- Methods of GenericServlet
- About init(ServletConfig),init() method

Working with welcome-file

- Configuring welcome-file in web.xml

Methods of loading Servlet

- About load on start up
- When client send first request

ServletContext

- What is ServletContext?, Need of ServletContext
- Methods of ServletContext
- Different ways of getting ServletContext object
- Context init parameters
- Defining Context init parameters in web.xml

HttpServlet

- What is HttpServlet?
- Methods of HttpServlet

- HTTP Request methods
- About public service and protected service methods
- About doXXX() methods
- Difference between GenericServlet and HttpServlet
- Diff Between doGet(-,-) and doPost(-,-) methods

HttpServletRequest

- HttpServletRequest VS ServletRequest
- How to read request parameters of HttpServletRequest
- How to read request headers of HttpServletRequest

HttpServletResponse

- HttpServletResponse VS ServletResponse
- About HttpServletResponse methods
- Response status codes

Html To Servlet Communication

Form Validations

Different Types of Form Components

Working With Multiple Hyperlinks

Working With Multiple Submit Buttons

War File Creation

Servlet To DataBase Software Communication

RequestDispatching

- What is RequestDispatcher?
- What is need of RequestDispatcher?
- RequestDispatching methods
- Include
- Forward

- Difference between include and forward methods
- Getting RequestDispatcher
- Using ServletRequest
- UsingServletContext
- Difference between getting RequestDispatcher using ServletRequest and ServletContext
- What is Servlet Collaboration?

Redirecting

- About sendRedirect method
- Difference between sendRedirect and forward methods
- setStatus, response.setHeader methods

Attributes

- What is Attribute
- Difference between parameter and attribute
- Scope of Attributes
- Request scope, Context scope
- Session scope
- Adding, removing and modifying attributes

Thread Saftey In Servlet Programming

State and Session Management

- About connectionless protocol
- About connection oriented protocol
- Stateless protocol
- What is state/session management
- Need of session management
- Session management methods
- URL rewriting, Hidden form fields

- Cookies, HttpSession

URL Rewriting

- What is URL rewriting
- URL rewriting techniques
- Disadvantages of URL rewriting

Hidden form fields

- What is hidden form fields
- Disadvantages of hidden form fields

Cookies

- What is Cookie,
- How to create Cookie
- Methods of Cookie
- Types of Cookies
- Non Persistent Cookie
- Persistent Cookie
- How to add cookie to response
- How to read cookie from request
- Deleting cookie from servlet
- Properties of Cookie
- Disadvantages of Cookies

HttpSession

- What is HttpSession
- Advantage of HttpSession
- How to create HttpSession object
- How to read HttpSession object from request
- Invalidating HttpSession

- Using invalidate method
- Using session config in web.xml
- By setting time
- HttpSession attribute

Error Handling in Servlet

- Configuring in web.xml
- Programmatically

Filters

- What is Filter?
- Need of Filter & Lifecycle of Filter
- Filter mapping in web.xml
- About FilterConfig
- Defining config parameters in web.xml
- About FilterChain interface
- Methods of FilterChain
- What is Filter chaining?
- Working with filter chaining

Listeners

- What is Listener?
- Usage of Listener
- About Event Delegation Model Architecture
- Types of Listeners
- RequestListener
- ServletContextListener
- HttpSessionListener
- Types of Attribute Listener

- Request AttributeListener
- HttpSessionAttributeListener
- ServletContextAttributeListener
- About Event classes

Authentication and Authorization(Security In Servlet Programming)

- Understanding Authentication Mechanism
- HTTP basic authentication
- HTTP Digest authentication
- HTTPS Client authentication
- HTTP form based authentication

Working Connection Pooling

- Connection pooling in difference servers
- Tomcat, Weblogic, Glassfish
- JBoss Server / WildFly
- Developing servlet using serverside connection pooling

Working with domains

- Creating domain in weblogic server
- Deploying webapplication using console
- Creating domain in glassfish server
- Deploying webapplication using console

Working with Servlet 2.5/3.0/3.1 features

- Async Servlet
- File Uploading and Downloading
- Non-blocking I/O
- HTTP protocol upgrade mechanism
- Different Types Of URL Patterns

- Dynamic Registration Of Servlet (Developing Servlet Program Without web.xml)

Annotations in Servlet Programming

- Introduction to Annotations
- Types of Annotations
- Annotations Vs web.xml

JSP

Introduction

- What is JSP?
- Advantages of JSP & Applications of JSP
- Difference between JSP & Servlet

JSP Basics

- JSP Life cycle, JSP Lifecycle methods
- jspInit, _jspService
- jspDestroy, Saving jsp file as a,
- Public resource, Private resouse
- JSP tags

JSP Tags

- three categories of tags
- scripting elements, directives
- standard actions

Scripting Elements

- What is scripting elements
- Types of scripting elements
- Declaration tag
- Expression tag
- Scriptlet

JSP implicit object

- What is jsp implicit objects
- Need of jsp implicit objects
- Implicit objects of jsp
- Request, Response, Page
- pageContext, out, session
- exception, application, config

Directives

- What is directive?
- Types of directives
- Page directive, Include directive
- Taglib directive

Standard Actions

- What is standard action
- Standard actions
- <jsp:include>
- <jsp:forward>
- <jsp:param>
- <jsp:useBean>
- <jsp:setProperty>
- <jsp:getProperty>
- Working with <jsp:include>
- Difference between <jsp:include> and <%@include> directive
- Working with <jsp:forward>

Java Bean

- What is java bean?

- Usage of bean
- Properties of bean

Using Java Bean in JSP

- About <jsp:useBean>
- Attributes of <jsp:useBean>
- Working with scope of bean object
- Using bean with <jsp:include> and <jsp:forward>
- Assigning values to bean using <jsp:setProperty>
- Reading values from bean using <jsp:getProperty>

Expression Language

- What is Expression Language
- Advantage of Expression Language
- Syntax of defining expression
- Basic Operators in EL
- Implicit objects In EL
- pageScope, requestScope
- sessionScope, applicationScope
- param, paramValues, header
- headerValues, initParam, cookie
- pageContext, Defining functions

Working JSTL

- What is JSTL?, Version of JSTL
- Classification of JSTL tags
- Core tags, Formatting tags
- SQL Tags, XML tags
- JSTL Functions

- How to use JSTL in webapplication
- Working with Core tags, Formatting tags & SQL tags
- Working JSTL functions

Custom Tags

- What is tag?, Java based tag
- Components of tag library
- The Tag Handler Class
- The Tag Library Descriptor File
- Imports a tag library (referencing URL of descriptor file)
- Defines tag prefix
- Uses tags, About tag handler
- Tag interface, SimpleTag
- TagSupport, BodyTagSupport
- SimpleTagSupport
- Developing tag using SimpleTagSupport class
- Lifecycle of custom tag
- Tag hander class with attributes
- Developing tag with body

MVC Architecture

- MVC Design pattern
- MVC-1 Page Centric
- Advantages & Disadvantages
- MVC-2

Developing Project using MVC

- Integrating JDBC , Servlets And JSP

Spring

About Spring 5.x Training

The Spring Framework is an open source application framework for Java. This framework has taken the Java software community by storm. Spring provided the technology to develop everything from small, stand-alone applications to large complex, enterprise systems out of simple POJOs (plain old Java objects).

In this class, students are exposed to the light-weight Spring container, configuration, foundational API, and general Spring architecture. Not just a class that focuses on theory, this course is loaded with practical labs and deals with configuration, maintenance and architectural issues. After taking the class, developers will immediately be able to utilize the Spring Framework in their new or existing applications.

Spring 5.x Course Prerequisite

- Students should have a good understanding of the Java programming language. A basic understanding of relational databases and SQL is very helpful. A basic understanding of XML is also useful. Students that have attended Intertech's Complete Java have the necessary background for this course.

Spring 5.x Training Course Objective

- Learn how to download, setup and configure the Spring Framework
- Explore the Spring Container and Modules
- Discover the Spring philosophies and principles and how they impact application development
- Understand dependency injection
- Learn aspect oriented programming and how it is used to provide cross cutting concerns
- See how to accomplish data access with Spring's DAO Module
- Understand how Spring deals with transaction management
- Examine Spring's unit testing framework

Spring 5.x Training Course Duration

- 60 Working days, daily one and half hours

Spring 5.x Training Overview

Spring Introduction

- Differences between programming language, software technology and framework
- Introduction to Spring Framework

- Evolution of spring Framework
- Modules of Spring in Spring 1.x,2.x,3.x,4.x and 5.x
- MVC Architecture
- Role of spring framework in MVC Architecture application development
- Definition of spring framework
- POJO Class, POJI, JavaBean, Component Class ,spring bean classes

Spring Core Module

- Introduction to IOC
- Introduction to Spring Container/IOC Container
- Types of Dependency Injections
- Setter injection
- Constructor injection
- Aware injection
- Method injection
- Lookup method injection
- Introduction to Design patterns
- Factory DesignPattern
- Strategy Design pattern
- Layered Application demonstrating real time dependency injection
- Resolving/identifying params in constructor injection
- Bean Inheritance
- Collection Merging
- Null injections
- Bean alias
- Default bean ids
- Performing dependency lookup by using IOC container

- <Idref>tag
- Understanding Factory Methods
- Factory Method Bean Instantiation
- Static factory method bean Instantiation
- instance factory method bean Instantiation
- Singleton java class and its usecases
- Bean Scopes
- Singleton
- Prototype
- Request
- Session
- Application
- Websocket
- Bean Wiring
- Explicit wiring
- Implicit wiring or auto wiring
- ByType
- ByName
- Constructor
- Autodetect
- p-namespace,C-namespace
- ApplicationContext Container
- Preinstantiation of singleton scope beans
- Working with properties file
- I18n (Internationalization)
- Event Handling

- BeanFactory Vs ApplicationContext
- Automatic registration of bean post processor and bean factory post processor
- Bean Life Cycle
- Declarative approach
- Programmatic approach
- Annotation driven approach
- Nested IOC Containers
- Presentation tier
- Business tier
- Various attributes of <ref> tag
- FactoryBean
- ServiceLocator as factory bean
- FactoryMethod bean Instantation based service locator
- Method Replacement/Method Injection
- Aware Injection
- Lookup method injection
- BeanPostProcessor
- BeanFactoryPostProcessors
- PropertyEditors
- Custom property editors
- Spring Expression Language(SpEL)

Spring Core Module with Annotations

- Spring stereo type annotations
- @Component,@Service,@controller,@Repositry and etc…
- @Autowired,@Qualifier,@Lazy and etc…
- Working with Java config annotations

- @Named,@Inject,@Resource and etc…
- Working with properties file in annotations environment
- Developing Layered applications in annotations environment

Spring Core Module with 100% Code/Java Config Approach.

- Working with @Bean, @Configuration,@Lazy,@PropertySource and etc…
- Developing Layered application
- Working with AnnotationConfigApplicationContext

Spring Boot Core

- Introduction to Spring Boot
- Spring Boot primary goals
- Spring boot features
- Spring Boot Starters
- Understanding @SpringBootApplication
- Auto configuration
- Example Applications
- Spring Profiles
- properties Vs application.yml
- Spring Boot Standalone flow
- Working with sts plugins in eclipse to develop spring boot application

Spring JDBC/DAO

- Introduction
- Plain JDBC limitations
- Spring JDBC/DAO Advantages
- Working with different Data Sources
- JdbcTemplate
- JNDI Registry and ServerManaged Jdbc connection pool

- Callback Interfaces
- Batch processing/Updating
- NamedParameterJdbcTemplate
- Working with SimpleJdbcInsert, SimpleJdbcCall
- SimpleJdbcCall to call PL/SQL procedures
- Mapping SQL operations as Sub Classes
- Spring JDBC/DAO with Annotations
- Spring JDBC/DAO with 100% Code Approach
- Spring Boot JDBC/DAO
- Spring Boot DAO-JdbcTemplate
- Spring Boot DAO-Simple Jdbc Insert
- Spring Boot Application Flow
- Working with DataSources through AutoConfiguration in Spring Boot 1.x and 2.x

Spring AOP Module

- Introduction
- Need of AOP
- Proxy design patterns
- AOP Terminologies/Principles
- Aspect, Advice, Joinpoint, Pointcut
- Target Class, Proxy Class, Weaing
- Types of Advices
- Before Advice, After Advice, Around Advice
- Throws Advice
- Types of Pointcuts
- Static pointcuts, Dynamic pointcuts
- Programmatic Spring AOP

- Declarative Spring AOP
- @AspectJ Style AOP support
- Spring AOP/AspectJ AOP with Annotations
- Spring AOP/AspectJ AOP with 100% Code Approach
- Spring AOP/Aspectj AOP with Spring Boot AOP

Spring Transaction Management

- Introduction to Transaction Management
- Local Transaction Vs Distributed Trasanction
- 2pc Principle
- Transaction models
- Flat Transaction Model
- Nested Transaction Model
- Need of Spring Transaction Management
- Choosing Spring Transaction Manager
- DataSourceTransactionManager
- Hibernate TransactionManager
- JTATransactionManager and etc…
- Different ways of implementing of Spring Transaction management
- Programatic approach
- Declarative approach using Spring AOP/AspectJ AOP
- Annotation Driven Approach using AspectJ AOP
- 100% code Driven approach using AspectJ AOP
- Spring Boot Driven approach using AspectJ AOP
- Transaction Attributes
- Transactions and integration testing
- Distributed TransactionManagement implementation using webLogic server,Atomikos API

- Configuring Transaction isolation Levels
- Read uncommitted, Read Committed
- Repeatable Read, Serializable
- Working with RollBackfor,noRollBackfor,Timeout and etc…in Transaction Management

Spring MVC

- Introduction To MVC
- Understanding MVC1,MVC2 Architectures
- Front Controller Design Pattern
- Intercepting Filter Vs Front Controller
- Different types of Servlet URL patterns
- Spring MVC Resources
- Spring MVC flow
- Structural Flow
- Strategy Flow(Code based Flow)
- DispatcherServlet
- Different Controller Classes
- ParamaterizableViewController
- UrlFileNameViewController
- AbstractController
- AbstractComandController
- SimpleFormController
- MultiActionController
- AbstractWizardFormController and etc…
- Developing Mini Project with CURD operation
- ContextLoaderListener
- Working with Two Containers

- HadlerMappings
- BeanNameUrlHandlerMapping
- SimpleUrlHandlerMapping
- ControllerClassNameHandlerMapping
- DefaultAnnotationHandlerMapping
- RequestMappingHandlerMapping and etc…
- HandlerMappingChaining
- Form Validations
- Enabling Server side Validations only when client side validation are not enabled
- <form:errors>
- ViewResolvers
- InternalResourceViewResolver
- UrlBasedViewResolver
- ResourceBundleViewResolver
- XmlViewResolver
- TilesViewResolver
- BeanNameViewResolver and etc…
- ViewResolverChaining
- Views
- InternalResourceView
- JstlView, TilesView, AbstractPdfView
- AbstractXlsView and etc….
- Exception Handling in Spring MVC
- Tiles integration with Spring MVC
- MessageSources and I18N
- Formatting Labels, Formatting Numbers

- Formatting Dates,
- Formatting Currency Symbols
- Locale
- MVC namespace
- Handler interceptors/Adapters
- Checking Browser Type
- Checking Timeout period
- Preventing double posting problem
- PDF Views and Excel Views
- File Uploading and Downloading
- Spring MVC with Annotations
- Annotation driven Controllers
- @RequestMapping, @Controller
- @ModelAttribute,@SessionAttribute,@RequestParam
- RequestToViewNameTranslator
- MVC NameSpace
- Annotation driven Form validation using Hibernate Validator API,JEE validator API
- Spring MVC with 100% Code Approach
- Dynamic Registration of Servlet
- WebApplicationInitializer
- SpringServletContainerInitializer
- @EnableWebMVC,@Import
- AbstractAnnotationConfigDispatcherServletInitializer
- Spring Boot MVC
- SpringServletInitializer
- Working with embedded TomcatServer

- Spring Boot MVC flow
- Spring Boot dev tools
- Properties,application.yml in Spring Boot
- Developing Mini Project with CURD operation
- Solving double posting problems in Spring Boot MVC using PostRedirectGetPattern
- Profiles in spring ,Spring boot

Spring Security

- Introduction
- Authentication authorization
- Authentication Manager and authentication info provider
- Need of Spring Security
- DeliagatingFilterProxy
- SecurityNameSpace
- Form Login
- Remember Me
- Session Concurrency
- Logout and etc…
- Working with Different Authentication Providers
- Xml File,Properties File,DataBase,LDAP Server
- Security Examples
- Using Xml Configuratuions
- Using Annotation Configurations
- 100% Code Driven Configurations
- Spring Boot Configuration
- Using LDAP Server as Authentication Provider

Spring ORM

- Introduction to ORM
- Spring ORM Advantage
- Integrating with Hibernate
- Spring with hibernate using HibernateTemplate
- HiberanteTemplate and its methods
- HibernateDAOSupport
- HibernateCallback interfaces

Spring Data and Spring Data JPA

- Need of Spring Data
- Spring Data JPA
- Finder Methods of Finder API
- Repositories
- JpaRepository
- CURDRepository
- Paging and Sorting Repository and etc…
- Spring Data Custom Query
- Automatic custom Query
- Manual custom Query(@Query)
- Spring Data Jpa Exception Translator
- Difference b/w Hibernate and Spring Data JPA
- Interacting with Mongo DB
- JUnit Test Cases

Spring Batch

- Need of Batch Processing
- Need of Spring Batch
- Understanding Spring Batch Architecture

- Working with Batch NameSpace
- Working with different ItemReaders, ItemWriters and ItemProcessors
- Converting Database Data to CSV File
- Converting CSV File to XML File
- Converting XML file to CSV File
- Spring Batch Application using Spring Boot
- JobBuilderFactory
- StepBuilderFactory
- Step point TaskLet and etc….
- Working with scheduler

Spring Mail

- Understanding Java mail API
- Understanding SMTP,POP3,IMAP Protocols
- Understanding Mail server and Mail Clients
- Understanding Email Message Structure
- Spring Mail abstraction over Java Mail
- Spring Mail using Spring Boot

Introduction to Spring MicroServices

How to Explain Project Architectures

- Servlet, Jsp Project Architecture
- Spring MVC Project Architecture
- Spring Boot with Micro service Project Architecture
- Adapting Agile Methodology

Developing Project using Spring.

Hibernate & JPA

About Hibernate Training

Hibernate is the most popular object-relational mapping framework for Java environments. Object relational mapping in large enterprise applications is difficult. In this class, students learn object-relational mapping concepts and the various issues and options available in Java to address object persistence. With these fundamentals, the course then explores the basics of Hibernate object persistence and configuration. It also digs into the details of Hibernate mapping, queries, transactions, and concurrency.

This course is loaded with lots of hands on examples and deals with maintenance and performance issues. After taking this class, developers will be able to build faster, more flexible and easier to maintain application persistence layers with the Hibernate framework.

Hibernate Course Prerequisite

- Students should have a good understanding of the Java Programming language.
- A basic understanding of relational databases and SQL is very helpful.

Hibernate Training Course Content Objective

- Understand the challenges of mapping objects to relational databases
- Learn the architecture of Hibernate
- Know how to setup and configure Hibernate for a Java Project
- Learn to map Java classes and object associations to relational database tables with Hibernate mapping files
- Study Hibernate’s strategies for mapping Java inheritance trees to relational database tables
- Learn the Hibernate Query Language and Criteria for retrieving Java objects
- Explore Hibernate’s Caching Architecture

Hibernate Training Course Duration

- Normal Track 45 Working days, daily one and half hour
- Fast Track 30 Working days, daily two hours

Hibernate Training Course Overview

Advantages of Hibernate compared to JDBC

Introduction

ORM (Object Relational Mapping)

Hibernate Resources

- Configuration file

- Mapping file
- Persistent class or POJO
- Client application.

Hibernate Architecture

Installation and Directory Structure

Hibernate Data Types

First Application using Hibernate

Hibernate API

- Configuration
- SessionFactory
- Session
- Transaction

Object Life cycle in Hibernate

- Transient object
- Persistent object
- Detached object

CRUD operations using Session methods.

- save, persist, SaveOrUpdate
- update, merge, delete
- load, get
- flush, evict, clear etc

Versioning

Primary key Generators

- Assigned
- Increment
- Sequence
- Hilo
- Seqhilo
- Identity
- Foreign
- Native
- UUID
- Custom generator

Hibernate Query Language (HQL)

Joins in Hibernate

Batch processing and Native SQL

Criteria API

Criteria with projections

Inheritance Mapping

- Table per class
- Table per sub class
- Table per concrete class

Component Mapping

Custom Mapping

Collection Mapping

- <list>
- <set>
- <map>
- <bag>
- Mapping array
- Sorting collections

Association Mapping

- one to one
- one to many
- many to one
- many to many
- Uni directional
- Bi directional
- Explanation on inverse and cascade attributes

Caching

- First level Cache(Session cache)
- Second level Cache(SessionFactory cache)
- Query level cache

Connection Pool

- Default connection pool
- Server supplied pool

- Third party vendor connection pool

Transactions and Concurrency

- Programmatic transactions with JTA
- Optimistic Concurrency control
- Pessimistic Concurrency control

Hibernate Pagination

Hibernate Filter

Hibernate Interceptor

Connecting with Multiple Databases(Oracle, HypersonicSQL)

Integrating Hibernate with Servlet

Integrating Hibernate with Struts

Working with Hibernate Annotations

IDE: Eclipse, Netbeans

Developing Project using Spring, JPA, Hibernate.

Web Services

XML Web Services Course Overview

The Java Web Services class teaches students how to build Web Services and Web Service clients using Java technologies. The class includes a introduction to XML namespaces, XML Schema, SOAP, and WSDL before exploring Web service client or server-side development in Java APIs and tools. Specifically, this class focuses on JAX-WS.

XML Web Services Training Prerequisite

- Students should have a good understanding of the Java programming language and a basic understanding of XML. Students that have attended Core Java and Advanced Java have the necessary background for this course.

XML Web Services Course Objective

- Understand how Web services related to Service Oriented Architecture.
- Become familiar with the pillar Web service specifications for XML, XML Schema, SOAP, WSDL and UDDI.

- Pick up design patterns and best practices for Web service interface documents.
- Experience the development of Java Web services using the JAX-WS API.
- See how WSDL and Schema elements map to Java objects.
- Recognize and understand the difference between RPC and Document styled services.
- Learn how to make and test Web services that are highly interoperable

XML Web Services Course Duration

- 45 Working days, daily one and half hours

XML Web Services Training Content

XML

Introduction to XML

- Evolution of XML
- Need of xml
- Workflow of xml

Fundamentals of XML

- Xml Declarations
- Elements
- Comments
- Processing instructions
- Doctype declarations
- Entities
- Namespaces
- Realtime Examples with scenarios

Document Type Definition’s (DTD’s)

- Validation
- Element & Attributes
- Entities

- Realtime Examples with scenarios

Xml Schema Definition(XSD)

- About XSD
- Difference between XSD & DTD
- Schema Declarations
- Data types
- Element declarations
- Complex type declarations
- Complex Content
- Simple Type Declarations
- Real-time Examples with scenarios

About Parser

- Need of parser
- Types of parsers
- Parsing approaches
- Real-time Examples with scenarios

Schema Validation

- About schema validation api’s
- JAXP DOM Parser
- JAXP SAX Parser
- JAXP Validation API
- Realtime Examples with scenarios

About XPath

- Need of XPath
- XPath Expressions
- Difference between XPath & DOM API

- JAXP XPath API
- JDOM XPath API
- Realtime Examples with scenarios
- About XSLT
- XSLT vs XML
- XSLT basics
- JAXP Transformation API
- Realtime Examples with scenarios

XML Object Bindings

- The need of object binding
- JAXB API(Covers 2.0)
- About Marshalling & Unmarshalling
- Binding with XML beans
- Real-time Examples with scenarios

Web Services

About Web Services

- Introduction
- SOA Architecture Principles
- Types of web services

About REST Web Service

- Introduction
- Rest principles & Terminology
- About JAX-RS API
- Annotation Inheritance with REST
- REST Client Design Scenarios
- REST Service Endpoint & URL Design Scenarios

- Realtime Examples with scenarios

About SOAP Web Service

- Introduction
- Anatomy of SOAP
- Structure of SOAP
- SOAP with HTTP
- SOAP-1.2 Features

About SAAJ

- Introduction
- Creating a SOAP Message
- Retrieving SOAP Message
- Sending SOAP Message
- Adding attachments to SOAP Message

About WSDL

- Introduction
- WSDL Structure (Covers 2.0)
- Message Exchanging Design Patterns & Modes

About JAX-WS API

- Introduction
- Fundamentals of JAX-WS
- Web Service meta data annotations
- XML Generation using JAXB Annotations
- JAX-WS Annotations
- Realtime Examples with scenarios

About Web services Security

- Introduction

- Need of security
- Types of securities
- Real time Examples with scenarios

About Apache-CXF

- Introduction
- High level architecture
- Real time Examples with scenarios

About Apache-AXIS

- Introduction
- High level architecture
- Realtime Examples with scenarios

Course Highlights: –

- All examples are covered with real time scenarios
- Covers development with Eclipse & NetBeans IDE's
- Web services testing and monitoring tools

Developing Project using Web services.

Spring Boot, Micro Services

Spring Boot, Micro Services Course Overview

Attend Spring Boot Training by Expert. Spring Boot is a powerful framework, used to build web applications quickly with less code. The Course will cover how to use Spring Boot to build the various projects with knowledge.

Pre-Requisites of the course

- Java programming language
- Web Development Experience HTML5, CSS3 and JavaScript

Spring Boot, Micro Services Training Contnet

Introduction to spring boot

- Types of software architectures
- SOA and Monolith Architecture
- Why Microservices
- Detailed MicroService Architecture

– App Layer

– Business Layer

- Enteprise Layer

– Infra Layer

- Need of Spring Boot
- Difference between Spring & Spring Boot
- Advantages with Micro Services

Building Spring Boot Application

- Normal Spring Manual Approach
- Maven Overview
- Spring Initializer
- STS
- Eclipse with STS Plugin
- Understanding the Spring Boot auto configuration

Rest Annotation with In Memory Database & CRUD Operations

- H2
- Derby
- HSQL
- Redis Cache
- PostMan or Swagger Overview

Rest Annotation with Relation DB

- MySql
- PostGresSQL

JPA Repository Concepts

- Crud Repository
- JPA Query Concepts
- NamedQueries
- QueryAnnotation
- AsyncResults
- Pagination and Sorting

Actuator Concepts

- Production Monitoring
- Health Check Concepts
- Security Measurements

Spring Boot Custom Logging

- Logging Level
- Patterns Changes
- Rolling Logs

Spring Boot Profile Components

- Introduction
- Multiple Properties
- YML File
- Command Line Runner Example
- Real time scenarios of components

Auto Configuration

- Introduction
- @Conditional Flow

- Customize conditional annotations
- Spring Boot built in conditional annotations

Thymleaf Concepts

- Introduction
- Example on Web Application
- Validatins on Web Applications
- Internalization i18n Concepts

Integration with Spring Web

- Using Spring Web MVC
- Using Spring Restful
- Need of embedded servers & customization

Spring Boot Security

- Basics
- Basic Authentication
- Form Based Authentication
- Authorization
- Role Based Access Control
- Attribute Based Access Control
- LDAP Based
- SSL Security
- TLS Security

Database Concepts

- Spring JDBC
- Database to CSV
- Spring Batch
- Flyway Database Migration

- Liquid Database Migration
- Flyway vs Liquid
- Hikari Connection Pool

Core Concepts

- Spring Boot AOP
- Spring Boot Cache
- Guava Cache integration
- Caffenine Cache
- EH Cache
- MultiResourceItemReader
- Spring MVC vs JAX-RS
- SprinBoot with Jersey
- Junit Integration
- Rest Integration Test Cases

Micro Services

- Micro Services Introduction
- Principle and Characteristics
- Use cases and Benefits
- Challenges
- Design standards
- Micro Services Communication

– Synchronous

– Asynchronous

- Pitfalls

Micro Services Design Considerations

- Micro Services per JVM?

- Micro Services share the data stores?
- Micro Services Transaction boundaries
- User Interfaces integration with Micro Services
- Challenges in Micro Services implementation

Spring Cloud

- Introduction
- Cloud Architecture
- Cloud application benefits

Spring Cloud Config

- Introduction
- Setup version control repository
- Integration with repository

Netflix

- Introduction
- Eureka Server & Eureka Client
- Feign Client
- Ribbon

Fault Tolerance Concepts

- Circuit Breaker Pattern
- Hystrics Concepts, Hystrix Dashboard

API Gateway

- Introduction to ZUUL
- Design standards
- Integration

Messaging Queue Concepts (CloudBus)

- Apache KAFKA

- RabbitMQ
- JMS

Oatuh2 Concepts

- Client Types
- Protocol End Points
- Grant Types
- Implantation with Token Based
- JWT Tokens

Swagger API

- Introduction
- Integration

Cloud Hosting

- Pivotal Cloud Foundry Account Setup
- Hosting to Pivotal
- AWS Account Setup
- Hosting to AWS
- Enabling cloud features like load balancing, security

Developing mega real time Project.