



BY NAGOOR BABU

SPRING COURSE MATERIAL

BY

NAGOOR BABU

{ SPRING }

VOLUME - 1 }

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING SYLLABUS

1. Introduction:

1. Enterprise Appl

2. Enterprise Application Layers

1. Presentation Layer
2. Business Layer
3. Data Access Layer

3. System Architectures.

1. 1-Tier Arch.
2. 2-Tier Arch
3. n-Tier Arch

4. Types of Enterprise Applications.

1. Web Applications
2. Distributed Applications

5. Modeled Arch.

1. Model-I Arch.
2. Model-II Arch.

6. MVC

7. Requirement to user Frameworks

8. Types of Frameworks

1. Web Frameworks
2. Application Frameworks

9. Differences between Spring and Struts, JSF

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

10.Spring History

11.Spring Modules

- 1.Spring1.x Modules
- 2.Spring2.x Modules
- 3.Spring3.x Modules
- 4.Spring4.x Modules
- 5.Spring5.x Modules

2.Steps To Prepare Spring Application[Core Module Application]:

1. Introduction

- 1.Download Spring Framework from Internet.
- 2.Provide Spring Setup in Eclipse IDE
- 3.Prepare Bean Class
- 4.Prepare Bean Configuration File
- 5.Prepare Test / Client Appl.

3.Core Module

1.Introduction

2.IOC Containers

1. BeanFactory
- 1.XmlBeanFactory
2. Resources

 - 1.ByteArrayResource
 - 2.FileSystemResource

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 3.ClassPathResource
- 4.InputStreamResource
- 5.UrlResource
- 6.ServletContextResource
- 7.PortletContextResource

2. ApplicationContext

- 1.ClassPathXmlApplicationContext
- 2.FileSystemXmlApplicationContext
- 3.WebXmlApplicationContext

3.Beans in Spring Framework

- 1.Beans Definition
- 2.Beans Configuration
 - 1.XML Based Configuration
 - 2.Annotation Based Configuration
 - 3.Java Based Configuration
- 3.Bean Scopes
 - 1.singleton Scope
 - 2.prototype Scope
 - 3.request Scope
 - 4.session Scope
 - 5.globalSession Scope
 - 6.application Scope
 - 7.webSocket scope

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

8. Custom Scopes in Spring Framework.

4. Bean Lifecycle

1. Bean Loading

2. Bean Instantiation

1. By Constructor

2. By Static Factory Method

3. By Instance Factory Method

3. Bean Initialization and Destruction

1. By Custom initialization and destruction methods.

2. By InitializingBean and DisposableBean callback interfaces.

3. By @PostConstruct and @PreDestroy annotations

5. Beans Inheritance

6. Nested Beans

7. BeanPostProcessor

4. Inversion Of Control[IOC]

1. Dependency Lookup

1. Dependency Pull

2. Contextualized Dependency Lookup

2. Dependency Injection

1. Constructor Dependency Injection

2. Setter Method Dependency Injection

3. Different Types of Elements Injection

1. User defined data types elements injection.

2. List types injection

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 3.Set types injection
- 4.Map Types Injection
- 5.Properties types Injection
- 6.Circular Dependency Injection

5.Name Spaces

- 1.P-Name space
- 2.C-Name Space

6.Beans Autowiring or Beans Collaboration

- 1.Autowiring and its Modes
 - 1.no
 - 2.byName
 - 3.byType
 - 4.constructor
- 2.Annotation Based Wiring
- 3.Autodiscovery or Stereo Types
- 4.Java based Autowiring[Java Based Configuration]

7.Method Injection

- 1.Lookup Method Injection
- 2.Arbitrary Method Replacement

8.Event Handling

- 1.ContextRefreshedEvent
- 2.ContextStartedEvent
- 3.ContextStoppedEvent
- 4.ContextClosedEvent

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com



BY NAGOOR BABU

- 5.RequestHandledEvent
- 6.Custom Events In Spring Framework

9.Bean Validations in Spring Framework

10.Internationalization in Spring Framework

11.Bean Manipulations and Bean Wrappers

12.Property Editors

- 1.ByteArrayPropertyEditor
- 2.ClassEditor
- 3.CustomBooleanEditor
- 4.CustomCollectionEditor
- 5.CustomNumberEditor
- 6.FileEditor
- 7.InputStreamEditor
- 8.LocaleEditor
- 9.PatternEditor
- 10.PropertiesEditor
- 11.StringTrimmerEditor
- 12.URLEditor
- 13.Custom Property Editors[User defined]

13.Profiling

14.Spring Expression Language[SpEL]

- 1.SpEL Expressions
- 2.SpEL Operators
- 3.SpEL Variables

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4.SpEL Method Invocations

5.SpEL Collections

5.Spring JDBC/DAO Module:**1. Introduction****2.DAO Definition****3.Advantages of DAOs****4.Drawbacks with DAOs****5.Guidelines to prepare DAOs****6.Pain JDBC Vs Spring JDBC****7.JdbcTemplate****8.NamedParameterJdbcTemplate**

1.Parameter values through Map

2.Parameter Values through SqlParameterSource

1.MapSqlParameterSource

2.BeanPropertySqlParameterSource

9.SimpleJdbcTemplate**10.DAO Support Classes**

1.JdbcDaoSupport

2.NamedParameterJdbcDaoSupport

3.SimpleJdbcDaoSupport

11.Spring Batch Updations or Batch Processing**12.Stored Procedure and Functions in Spring JDBC**

1.Procedures and Functions with out CURSOR Types

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Procedures and Functions with CURSOR Types

13. Blob and Clob processing in Spring JDBC

1. AbstractLobCreatingPreparedStatementCallback
2. AbstractLobStreamingResultSetExtractor
3. LobCreator
4. LobHolder

14. Connection Pooling in Spring JDBC

1. Default Connection Pooling Mech.
2. Third Party Connection Pooling Mechanisms
 1. Apache DBCP
 2. C3P0
 3. Proxool
3. Application Servers provided Connection Pooling Mechanism
 1. Weblogic12c provided Connection Pooling Mechanism.

5. Spring ORM

1. Introduction

2. Hibernate Integration with Spring

1. Hibernate Introduction
2. Hibernate Application Development
3. Spring with Hibernate Integration.

3. JPA Integration with Spring

1. JPA Introduction.
2. JPA Application development

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3.Spring with JPA Integration.

4.iBatis integration with Spring

1.iBatis Introduction.

2.iBatis Application Development.

3.Spring with iBatis Integration.

6.Aspect Oriented Programming [AOP]

1.Introduction

2.AOP Terminology

1.Aspect

2.Advice

3.JoinPoint

4.Pointcut

5.Introduction

6.Target

7.Proxy

8.Weaving

9.Advisor

3.Types of AOPs

1.Proxy Based AOP

2.Declarative Based AOP

3.Annotation Based AOP

4.Advises

1.Before Advice

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 2.After Advice
- 3.After-Returning Advice
- 4.Around Advice
- 5.After-Throwing Advice

5.Pointcuts

- 1.Static Pointcut
- 2.Dynamic Pointcut.

7.Spring Transactions

- 1.Introduction
- 2.Transaction Attributes
- 3.Isolation Levels
- 4.Programmatic Based Transactions
- 5.Declarative Based Transactions.
- 6.Annotation Based Transactions

8.Spring web MVC Module

- 1.Introduction
- 2.Spring MVC Flow
- 3.Controllers
 - 1.Abstract Controller
 - 2.ParameterizableViewController
 - 3.MultiActionController

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4.Command Controllers

- 1.AbstractCommandController
- 2.AbstractFormController
- 3.SimpleFormController
- 4.AbstractWizardFormController

4.Handler Mappings

- 1.BeanNameUrlHandlerMapping
- 2.SimpleUrlHandlerMapping

5.HandlerInterceptor**6.ViewResolvers**

- 1.AbstractCachingViewResolver
- 2.XmlViewResolver
- 3.ResourceBundleViewResolver
- 4.UrlBasedViewResolver
- 5.InternalResourceViewResolver
- 6.VelocityViewResolver / FreeMarkerViewResolver

7.Spring Exception Handling**8.File Uploading and File Downloading****9.Internationalization****10.Spring MVC with Tiles****CONTACT US:**

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

INDEX

1. Spring Core Module.....	Page 014 to Page 235
2. Spring JDBC	Page 236 to Page 313
3. Spring AOP.....	Page 314 to Page 360
4. Spring ORM.....	Page 361 to Page 403
5. Spring Transactions.....	Page 404 to Page 425
6. Spring MVC.....	Page 426 to Page 696
7. Maven Tool.....	Page 697 to Page 741
8. Log4J Tool.....	Page 742 to Page 771

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGASHFT

{CORE MODULE}

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgashft.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING CORE MODULE INDEX

1. Spring Introduction.....	PAGE 017
2. Steps to prepare First Spring Application.....	PAGE 036
3. IOC Containers.....	PAGE 051
4. Beans In Spring Framework.....	PAGE 060
5. Inversion Of Control[IOC].....	PAGE 102
6. Beans Autowiring/Beans Collaboration.....	PAGE 130
7. Bean Validations in spring.....	PAGE 174
8. Event Handling.....	PAGE 181
9. Internationalization in SPRING.....	PAGE 192
10.Bean Manipulations and Bean Wrappers.....	PAGE 200
11.Property Editors.....	PAGE 209
12.Profiling.....	PAGE 216
13.Spring Expression Language [SpEL].....	PAGE 221

DURGATHA

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING

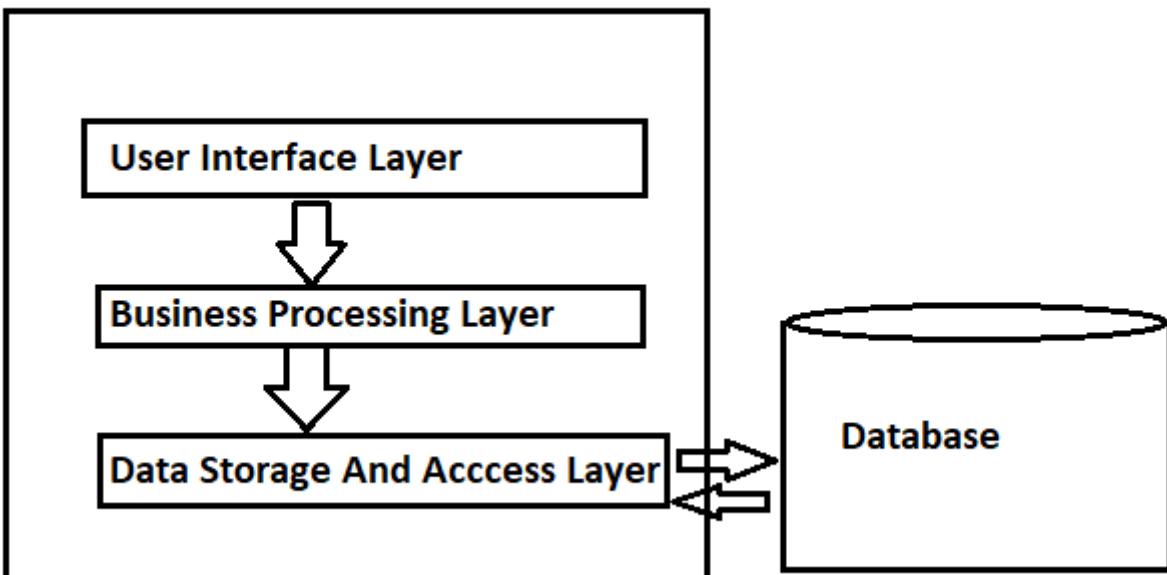
Introduction:

Enterprise: It is a business organization, it is a group of organizations running under single label.

Enterprise Application: It is a software application prepared for an enterprise in order to simplify their business processing.

To prepare Enterprise Applications, we have to provide the following three layers.

Enterprise Application Layers



1. User Interface Layer:

1. This layer is the topmost layer in enterprise application.
2. It will provide starting point to the users in order to interact with enterprise application.
3. It will provide very good environment to get data from users to submit data to server side application.
4. It will provide very good environment to perform client side data validations with java script functions.
5. User interface will provide very good environment to send different request types from

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

client or server like GET, POST, HEAD,

6.To prepare User Interface layer, we have to use a separate logic called as "Presentation Logic".

7.In Enterprise Application development, to prepare presentation logic we have to use the technologies like AWT, SWING, html, JSPs, Velocity, Free Marker, OGNL,

2. Business Processing Layer:

1. It is heart of the enterprise application, it can be used to define and execute all business rules and regulations which are required by the clients actually.
2. In Enterprise Application development, to prepare Business Processing layer we have to use a separate logic called as "Business Logic".
3. To provide Business Logic we have to use the technologies like Servlets, EJBs, DAOs,

3. Data Storage And Access Layer:

1. This layer is bottom most layer in enterprise applications, it will provide very good environment to interact with databases in order to perform persistence operations.
2. To prepare this layer in enterprise applications we have to use a separate logic called as "Persistence Logic".
3. To provide persistence logic we have to use a set of technologies like JDBC, EJBs-Entity Beans, Hibernate ,....

System Arch.:

To define level or height of the enterprise applications we have to use System Arch.

EX:

- 1-Tier Arch.
- 2-Tier Arch.
- 3-Tier Arch.
- n-Tier Arch.

1-Tier Arch.

1. In 1-Tier Arch, we have to prepare and execute the complete enterprise application with

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

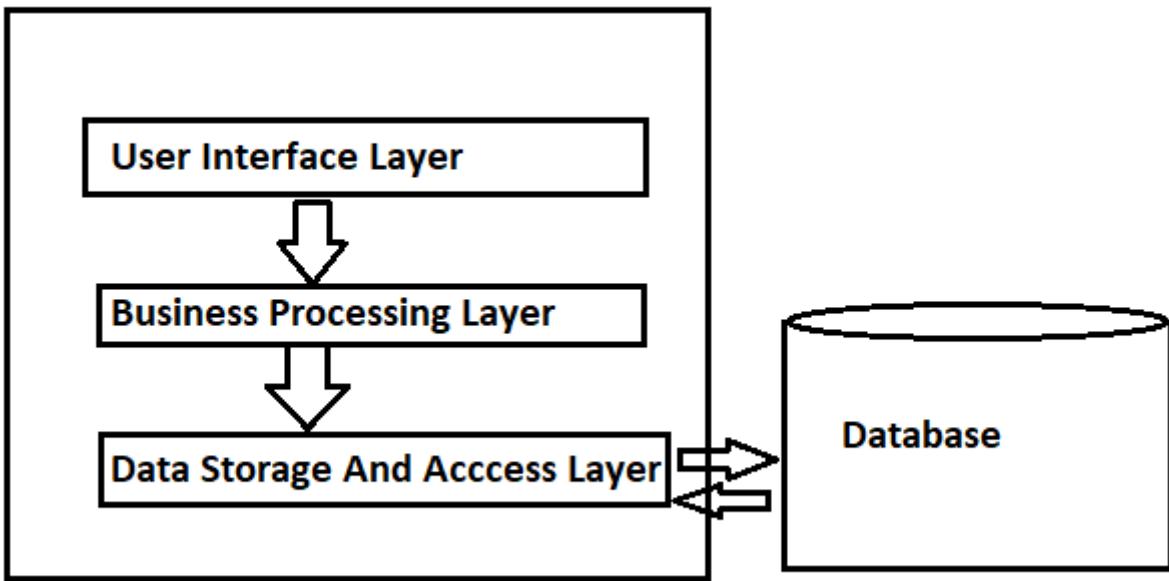


BY NAGOOR BABU

in a single machine.

2. In 1-Tier Arch, all the enterprise application layers like presentation Layer, Business Layer and Persistence layer must be provided in a single machine, no separation between all the layers, it will provide tightly coupled design in enterprise applications, it is not suggestible.
3. In 1-Tier arch, the complete enterprise application must be executed in a single machine, where single machine resources may not be sufficient to manage the complete enterprise application, it may affect the application performance.
4. 1-Tier Arch is suggestible for Standalone Applications, not for enterprise Applications or distributed Applications.
5. in 1-Tier Arch, Sharability and Reusability are very less , it may increase application length.
6. In 1-Tier Arch, multi user environment is not existed, only Single user environment is existed.

Machine



2-Tier Arch:

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

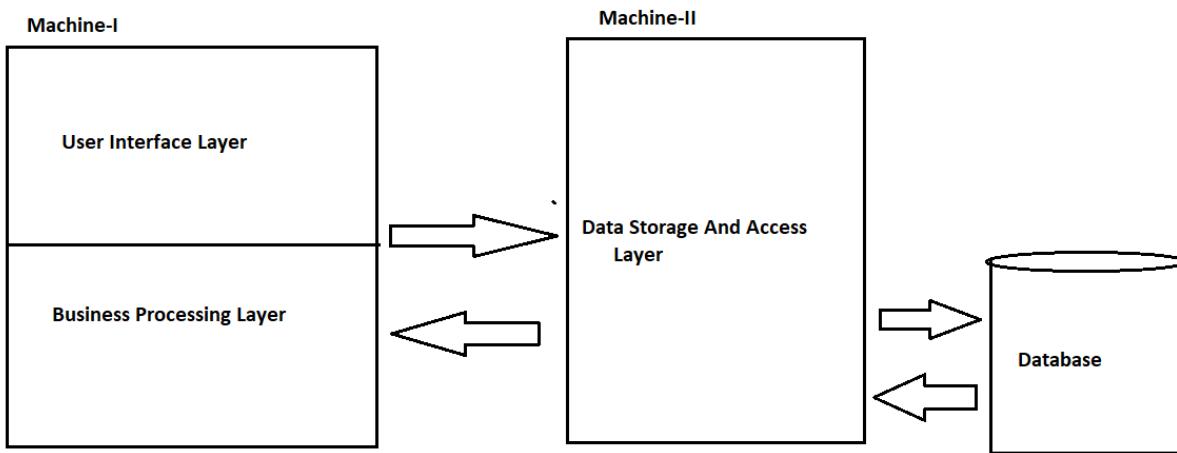
FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. IN 2-Tier Arch, we have to distribute the complete enterprise application over two machines.
2. In 2-Tier Arch, Tier-1 machine is able to manage presentation logic and Business Logic, Tier-2 machine is able to manage Persistence Logic.
3. In 2-Tier Arch, we are able to get loosely coupled design when compared with 1-Tier Arch, because, Persistence layer is separated from Presentation layer and Business Layer.
4. In 2-Tier Arch, database components are shared to client applications, so that, sharability and Reusability are increased.
5. 2-Tier Arch will provide multi user environment to access application.

Note: If want to use 2-Tier arch for web applications then we have use Tier-1 is for client , it has to manage presentation layer and Tier-2 is for Server , it has to manage Business Layer and Persistence Layer.



3. 3-Tier Arch:

- > This arch will propose to use three machines in order to execute the complete enterprise application.
- > In 3-Tier Arch, we will provide Presentation Layer at Tier-1 Machine, Business Layer at Tier-2 machine and Persistence layer at Tier-3 machine.
- > 3-Tier arch will provide more more loosely coupled design to design applications. -
- > 3-Tier Arch will provide Multi User environment, it will improve sharability and

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

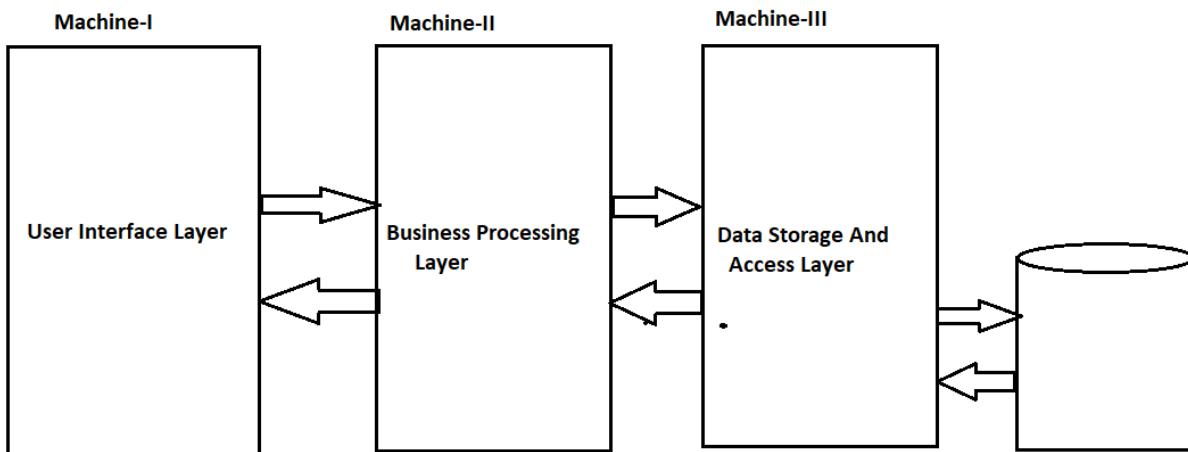


BY NAGOOR BABU

Reusability.

--> 3-Tier arch will improve application Server components sharability and database components sharability.

Note: If we increase no of Tiers in applications then flexibility will be increased, but, maintenance cost will be increased. In enterprise applications we have to increase no of tiers as per the purpose only, we must not increase no of tiers unnecessarily.



There are two types of Enterprise Applications.

- 1.web applications[Web Related Distributed Appl]
- 2.Distributed Applications[Remote based Distributed Appl].

Q) What are the differences between Web Applications and Distributed Applications?

Ans:

1. Web Application is a Client-Server Application, where the complete application logic is distributed over Server machine.

Distributed Application is a client-server Application, where the complete application logic is distributed over Client machine and Server machine.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. To prepare web applications, we will use a set of technologies called as web technologies.

EX: Servlets, JSPs,

To prepare Distributed Applications, we will use a set of technologies called as Distributed technologies.

EX: Socket programming

RMI

CORBA

EJBs

Web Services

3. The main intention of Web applications is to generate dynamic response from server.

The main intention of Distributed Applications is to establish Communication between local machine and remote machine in order to get Remote Services from Remote machine.

4. Web applications are executed by both web servers and application servers.

Distributed applications are executed by only application servers.

5. Web application is the collection of web components like servlets, jsp,... , which are executed by web containers.

Distributed Application is the Collection of distributed components like EJBs, which are executed by EJB Container.

6. In web applications, Client is fixed, that is, Browser.

In Distributed Applications, Client is not fixed, it may be a normal java program with main() method, it may be a GUI Application, it may be a Servlet program, it may be a JSP program,.....

To prepare web applications, SUN Microsystems has provided the following Modeled Arch.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 1 .Model-I Arch.
2. Model-II Arch.

1. Model-I Arch.

In Model-I Web Application Arch, we will use a JSP page as controller as well as Presentation part and a Java Bean component is acting as Model Component.

In Model-I web application Arch, a JSP page is acting as Controller to control the complete web application, so that, Model-I Web application Arch is also called as "Page-Centric Arch".

In Model-I web application Arch, a JSP page is taking responsibility to take requests from client, so that, Model-I web application Arch is also called as "JSP Front".

IN Model-I web application Arch, we will use JSP pages as Controller and for presentation, there is no clear cut separation between Controller logic and presentation logic, it will provide tightly coupled design in web applications, it is not suggestible in web applications.

In Model-I web applicatio Arch., a JSP page is acting controller, to perform controller functionalities the existed JSP features are not sufficient, where it is required to write java code inside JSP pages, it is against to JSP rules and regulations.

In Model-I web applicatio Arch., a JSP page is acting controller, to perform controller functionalities the existed JSP features are not sufficient, where it is required to write java code inside JSP pages, it is against to JSP rules and regulations.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

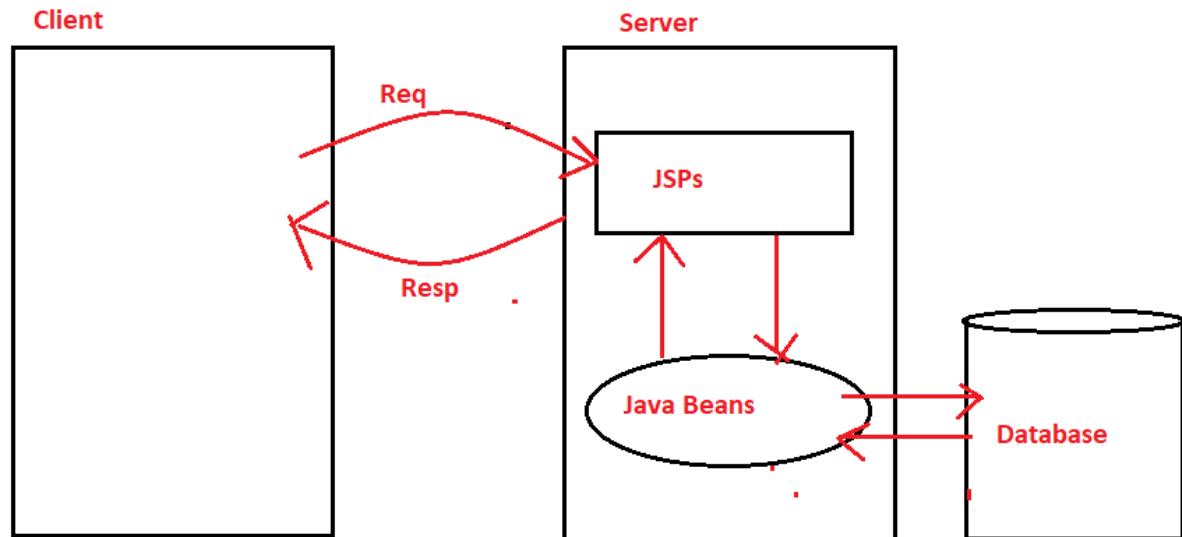
Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU



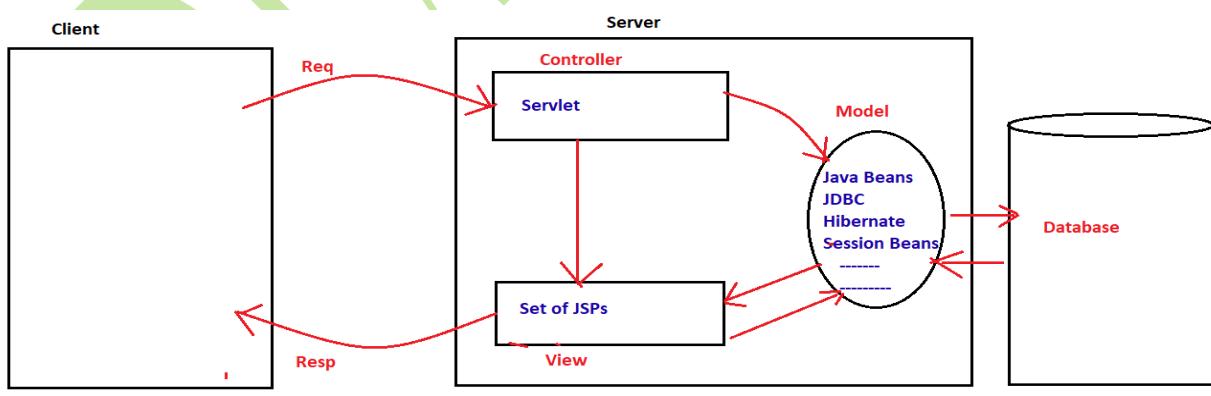
Model-II Arch :

In Model-II web application Arch, we will use a Servlet as controller, a set of JSP pages as View part and Java bean, DAO, JDBC,... as used as Model Components.

In Model-II Arch, a servlet is acting as controller to control the complete web applications, so that, Model-II web Arch is called as "Servlet-Centric Arch".

In Model-II Arch, a servlet is taking responsibility to take all the requests from Client , so that, Model-II Arch is also be called as "Servlet Front Arch".

Note: Model-II Arch is an implantation of MVC Arch, on the basis of Model-II Arch only the web frameworks like Struts, JSF,... are designed.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Rules and regulations of MVC Arch:

1. MVC is a design pattern, it will define a standard template to prepare web applications.
2. MVC will define standard flow of execution to prepare web applications.
3. In MVC based web applications, we must use a Servlet as controller and a set of JSP pages as view part.
4. In MVC based web applications, we must provide single controller per application.
5. In MVC based web applications, Controller component must take all the requests which are coming from clients and View part must take the responsibility to generate response to client.
6. IN MVC based web applications, both controller and view part are not responsible to interact with database, they have to interact with database through Model component.
7. In MVC based web applications, Controller is able to set data to model component, not to get data from model component and View part is able to get data from model component , not to set data to Model component.
8. IN MVC based web applications, we can provide any no of pages as view part, but we must provide all the pages as Java code less.
9. IN MVC based web applications, we can provide no of pages as view part, where we must not provide page-to-page communication directly, where we have to provide page-controller-page communication.

In general, from application to application, some components like ControllerServlet and some generic Services like Internationalization, Security, Data Validations,... are very much common, these common components may provide 70% implementation in the complete enterprise application.

CONTACT US:Mobile: +91- **8885 25 26 27**+91- **7207 21 24 27/28**US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In the above context, If any third party organization is providing the common 70% implementation then developers may take responsibility to provide the remaining 30% of the implementation.

In the above situation, some third party organizations like Apache Software Foundations, Soft Tree, has provided the common 70% of the implementation in the form of their own products called as "Frameworks".

Framework is a pre fabricated Software components that programmer can reuse, share and customize inorder to simplify enterprise application development.

Framework is a semi implemented application, it will provide very good environment to prepare enterprise applications as per developers convenience.

Framework is the Collection of Tools and APIs, it will provide very good environment to prepare enterprise applications in simplified manner.

In enterprise application development, Frameworks wil provide the folowing advantages.

1. Frameworks will define standard template to design applications.
2. Frameworks will define a fixed flow of execution between the components.
3. Frameworks provide parallel development and modularization.
4. Frameworks will provide all the commonly used generic services like I18N, Security, Exception handling, Data validations,....
5. Frameworks will reduce development time.
6. Frameworks will reduce application development cost.
7. Frameworks will increase productivity.

There are two types of Frameworks .

1. Web Frameworks
2. Application Frameworks

1. Web Frameworks

Web frameworks will provide environment to design and execute only web applications.
EX: Struts, JSF, Xwork2,

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Application Frameworks

Application Frameworks will provide very good environment to prepare all the types of applications like Standalone Applications, Web Applications, Distributed Applications,.....

EX: Spring

Q) What are the differences between Struts, JSF and Spring?

Ans:

1. Struts and JSF are web frameworks, which will provide very good environment to prepare and execute web applications only.

Spring Framework is an application Framework, it will provide very good environment to prepare and execute all the types of applications like standalone applications, web applications, distributed applications.....

2. Struts and JSF are designed on the basis of only MVC design pattern.

In Spring , only WEB Module is designed on the basis of MVC, Spring is using no of other design patterns like IOC[Dependency Injection], Locator design patterns, Creational Design Patterns, Decorator Design pattern,.....

3. Struts is controller layered framework, it has very good focus on controller layer in MVC.

JSF is view layered Framework, it has very good focus on View layer in MVC. Spring Framework has provided very good support for all Controller Layer, Model Layer and View Layer.

4. In Enterprise Application Development, Struts and JSF are used to prepare mainly Presentation layer.

In Enterprise Application Development, Spring Framework will cover all the layers like Presentation Layer, Business Layer, Persistence Layer.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

5. Struts and JSF are provide support for the basic services like I18N, Validations, Exception Handling,..., but, Struts and JSF are not providing support for the middleware services like JAAS, JNDI, JTA, Java Mail, JMS,.....

Spring Framework is providing very good support for all basic services like I18N, Validations, Exception Handling ... and the Middleware services like JAAS, JNDI, JTA, Java Mail, JMS, JCA.....

6. Struts and JSF are not modularized Frameworks, to prepare any application in Struts and JSF we have to load all the jar files irrespective of their utilization.

Spring framework is modularized framework, to prepare applications in spring framework we will load only the module respective jar files, not required to load other modules respective jar files unnecessarily.

7. Struts and JSF are more API dependent; they are not having POJO/POJI kind of implementations.

Spring is less API dependent, it has POJO/POJI kind of implementations.

8. Due to the above reasons, debugging and testing are difficult in Struts and JSf.

Testing and Debugging are very simple in Spring framework.

9. Struts and JSF are heavy weight Frameworks.

Spring is light weight Framework.

10. Struts and JSF are not providing any predefined support to integrate the other applications like JDBC, EJBs, Hibernate, JPA, RMI,....

Spring has provided very good predefined support to integrate the other applications like JDBC, EJBs, Hibernate, JPA, RMI,....

11. Struts and JSF are allowing the basic view related technologies like Html, JSp, to

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

prepare view part.

Spring Framework is allowing the most advanced view related tech like Velocity, Free marker,... along with basic view related tech in order to prepare view part.

12. Struts and JSF are not having Aspect Oriented Programming to prepare applications.

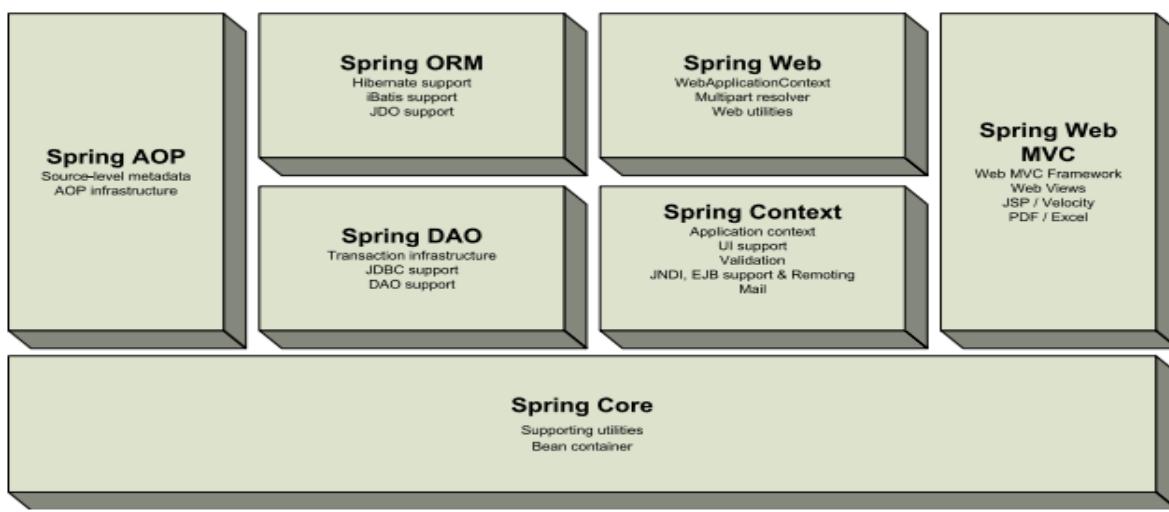
Spring has provided very good Aspect Oriented Programming to prepare Applications.

Spring History:

- 1.Home: Interface1
- 2.Author: Mr. Rod Janson
- 3.Objective: To simplify and Accelerate the complete Enterprise Applications.
- 4.Type: Open Source Software.
- 5.Type of Framework: Application Framework.
- 6.Initial Version: Spring1.0[Oct, 2004]
- 7.Used Version: Spring3.x[Dec, 2014]
- 8.Latest Version: Spring4.3.8[April, 2017]
- 9.Website: <http://spring.io>
- 10.Designed on: Java[JAVA SE API, Servlets API]
- 11.Compatibility: Supported by All IDEs and all Servers
- 12.Design Tool: STS[Spring tool Suit][Designed on the top of Eclipse]

Spring Modules:

Spring1.x version:



CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

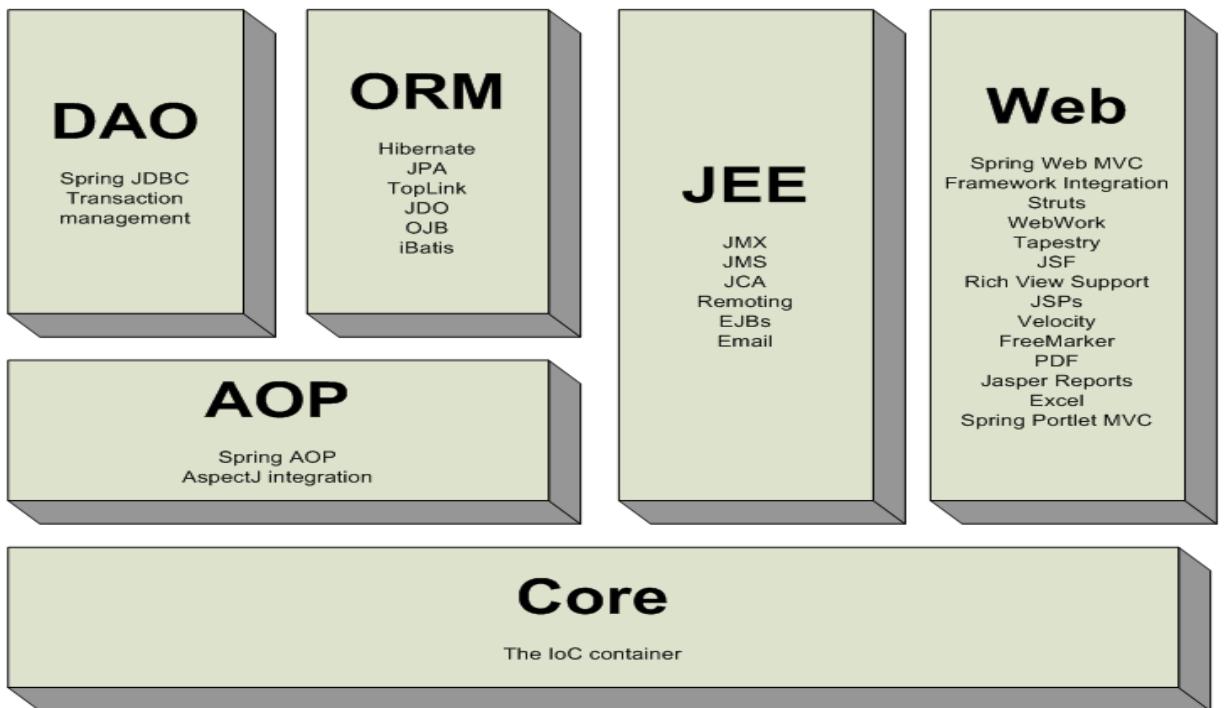
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

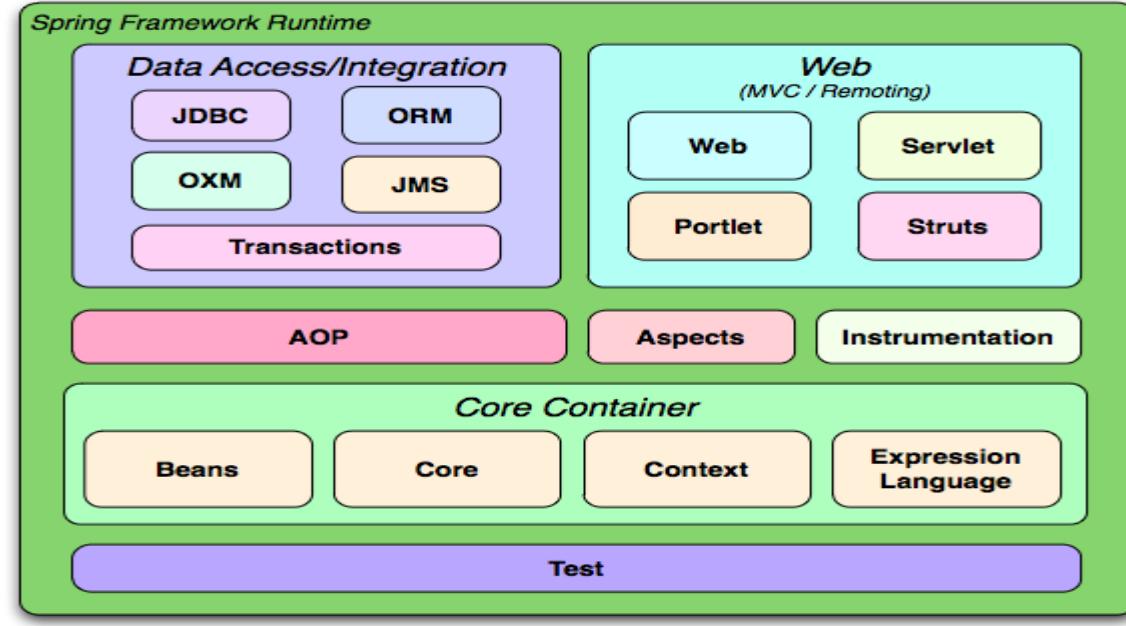


BY NAGOOR BABU

Spring2.x version:



Spring3.x Version:



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

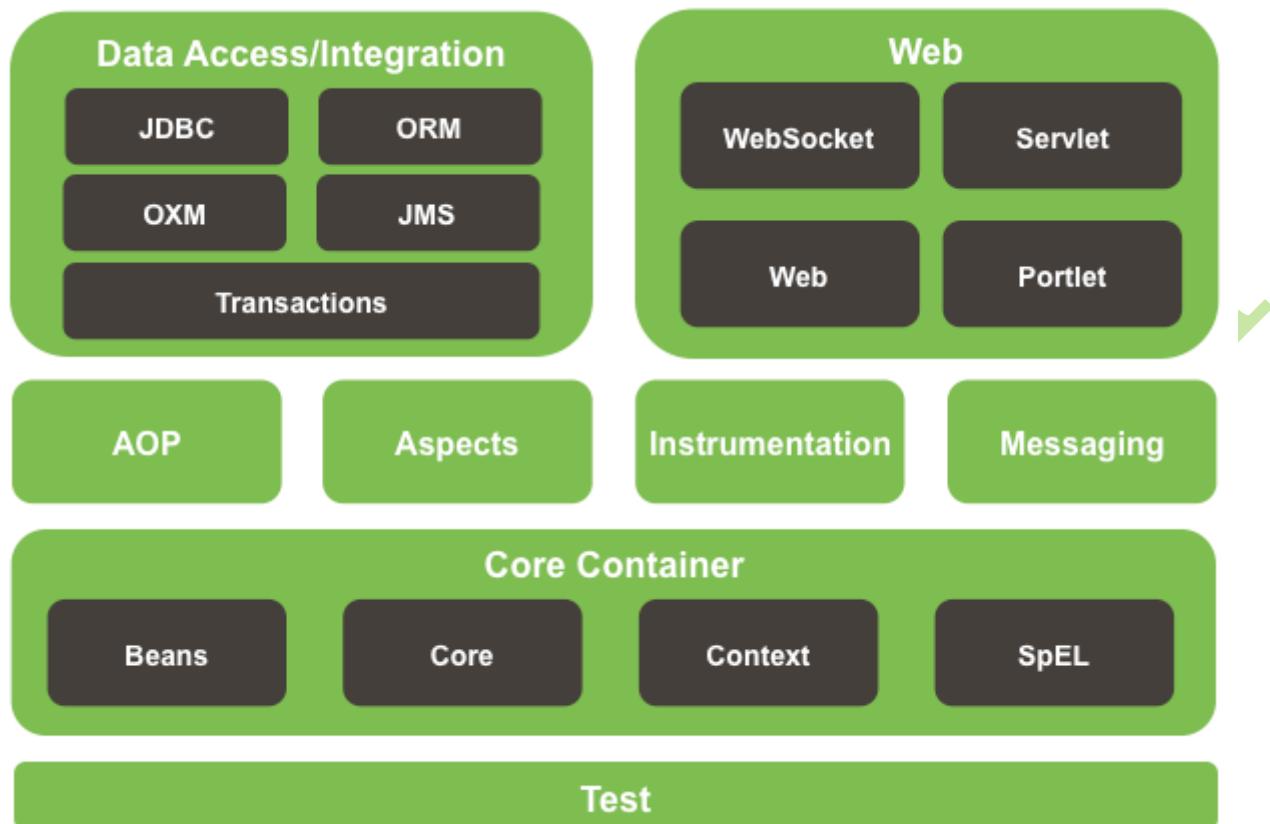
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

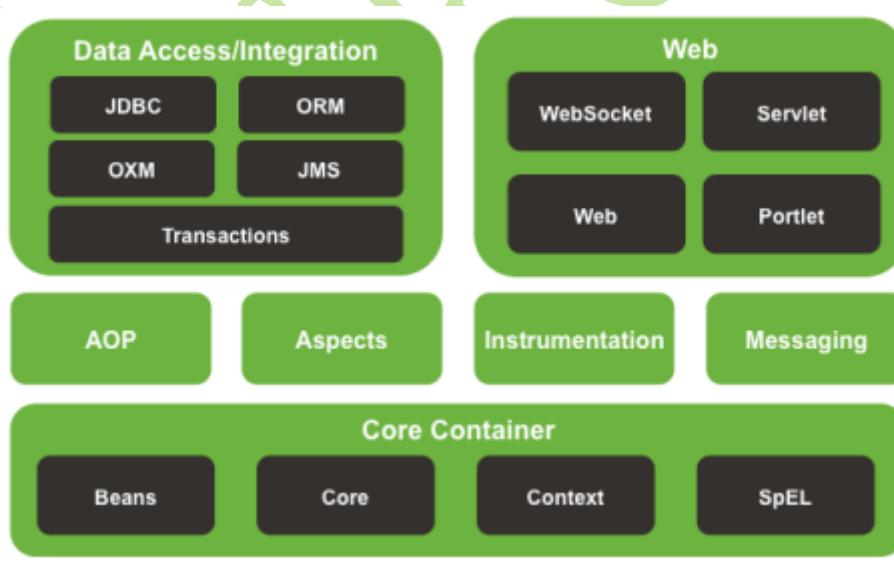


BY NAGOOR BABU

Spring4.x Version:



Module5.x Version



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Core Module:

It is fundamental module in Spring Framework; it has provided basic foundation for all other modules of spring framework.

This module can be used to prepare Standalone Applications directly.

This module is able to provide the features like IOC Containers, Beans, Dependency Injection.....

AOP Module[Aspect Oriented Programming]:

IN general, if we prepare enterprise applications by using only Object Orientation then we have to provide both business logic and Services like Transactions, JMS, JAAS,... in combined manner, it will provide tightly coupled design, it will provide less sharability and less reusability.

In the above context, to improve sharability and Reusability we have to provide loosely coupled design , to get loosely coupled design we have to apply Aspect Oriented Programming.

In Aspect Oriented Programming, we will declare each and every service as an aspect and we will inject these aspects in Business Logic at runtime.

JDBC/DAO Modules:

The main intention of this module is to interact with database from Spring application inorder to perform database operations with JDBC Persistance mechanism.

JDBC/DAO modules are able to abstract common JDBC implementation inorder to simplify Database interaction from spring applications by providing template classes.

EX:

In JDBC, if we want to interact with database then we have to use the following steps.

1. Load And Register Driver

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX: Class.forName("oracle.jdbc.OracleDriver");

2. Establish Connection between Java appl and DB.

EX:

```
Connction con=DriverManager.getConnection("jdbc: oracle:thin:@localhost:1521:xe",
"system", "durga");
```

3. Create Statement/PreparedStatement/ CallableStatement as per the requirement.

EX: Statement st=con.createStatement();

4. Write and Execute SQL Queries

```
ResultSet rs=st.executeQuery("select * from emp1");
```

5. Close the Connection

EX: con.close();

In the above JDBC Steps, Load and Register Driver, Establish Connection and Create Statement and close the Connection are vary common in all the JDBC applications, here Spring JDBC/DAO modules are able to abstract the commonly used steps and giving option to the developers to provided variable part , that is, Executing Sql queries .

Spring DAO and JDBC modules are having their own Exception Classes hierarchy to expose the exception details.

Spring DAO/JDBC modules are converting the JDBC generates checked exceptions to Spring defined Unchecked Exceptions by using Exceptions Re-Throwing Mechanism.

EX:

```
try{
---Exception-1 ----JDBC Checked Exception
}
catch(Exception_Name e){
---Exception-2---Spring Unchecked Exception
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

ORM Module[Object-Relational Mapping]:

ORM is the mapping between a table, properties or columns and primary key column with the respective bean component provided Bean class name, ID property, normal properties,

ORM has define a set of rules and regulations to provide mapping between Object Oriented Data Model and Relational Data Model in order to achieve data persistency.

EX: Hibernate, JPA, Toplink,....

To prepare Hibernate Application then we have to use the following instructions.

1. Create Configuration class object.

EX:

```
Configuration cfg=new Configuration();
cfg.configure();
```

2. Create SessionFactory class object:

EX:

```
SessionFactory sf=cfg.buildSessionFactory();
```

3. Create Session object:

EX:

```
Session s=sf.openSession
```

4. Perform Persistance operations:

EX:

```
Object obj=s.load("com.durgasoft.Employee.class", 111);
```

5. Close SessionFactory and Configuration:

EX:

```
sf.close();
cfg.close();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In the above Hibernate Application steps, Create Configuration class object, Creation SessionFactory object , create Session object and close SessionFactory and Configuration steps are very common in all hibernate applications, in this case, Spring ORM module is able to abstract all the common instructions in order to simplify Data persistence in enterprise applications.

Spring ORM module has provided its own Exceptions hierarchy to expose behalf of the under lying Persistence mechanism like Hibernate, JPA,.... provided checked exceptions.

JAVA EE/Context/Remoting:

The main intetion of this module to integrate Spring applications with the diistributed tech applications like EJBs, RMI,.... it able to get moddleware services like JNDI, JTA, JAAS,.... from J2EE .

WEB/WEB-MVC Modules:

WEB Module has provided very good environment to integrate other MVC based framework applications like Struts, JSF, XWork2,.....

WEB-MVC is an MVC implementation provided by Spring framework directly inorder to prepare web applications.

Test:

Spring Framework has provided its own testing environment to test the enterprise applications by the developers[Unit Testing] in their own way in the form of Test module.

Note: Spring framework has provided Test module right from its Spring3.x version.

Instrumentation:

Spring Framework has provided Instrumentation API to perform manipulations over the actions which are generated from the .class files with out having modifications over the source code.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Messaging:

Spring Framework has provided Messaging module along with its Spring4.x version to provide Messaging Services what JMS is providing previously as part of J2EE.

Note: When we perform any transactions in bank application , we are able to receive mobile updations, in this context, if our mobile device is not ready to receive messages WEB Module has provided very good environment to integrate other MVC based framework applications like Struts, JSF, XWork2,.....

then we have to keep the respective message in Messaging Container or JMS Server to send the respective message when the target machine / device is ready.

Steps to prepare First Spring Application

1. Download Spring Framework from Internet.
2. Provide Spring Setup in Eclipse IDE
3. Prepare Bean Class
4. Prepare Bean Configuration File
5. Prepare Test / Client Appl.

1. Download Spring Framework from Internet.

- a) Download spring jar files in the form of "spring-framework-4.1.6.RELEASE-dist.zip" from the URL "<https://repo.spring.io/release/org/springframework/spring/4.1.6.RELEASE/>" and "commons-logging-1.2.jar" from apache website.
b) Unzip "spring-framework-4.1.6.RELEASE-dist.zip" at a particular location in our machine[E:\softwares\spring]

2. Provide Spring Setup in Eclipse IDE

- a) Open Eclipse IDE.

1. Download eclipse-jee-neon-2-win32.zip file from www.eclipse.org website.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Extract ZIP file at "C" drive and get "eclipse" folder.
3. Click on "eclipse" icon at "C:\Eclipse" location
4. Provide Workspace "D:\spring3\core\eclipse".
5. Click on "OK" button.

b) Create Java Project

1. Right Click on "Project Explorer" Browser.
2. Select "New"
3. Select "Project".
4. Select "Java Project".
5. Click on "Next" button.
6. Provide project name "app1".
7. Click on "Next" button.
8. Click on "Next" button.
9. Click on "Yes" button.

c) Create User defined Library with all Spring jar files and add that User defined Library to "Java project".

1. Right Click on Project [app1].
2. Select "Properties".
3. Select "Java Build path".
4. Select "Libraries".
5. Click on "Add Library" button.
6. Select "User library".
7. Click on "Next" button.
8. Click on "User Libraries" button.
9. Click on "New" button.
10. Provide Library Name "Spring4.1.6_Lib_New".
11. Click on "OK" button.
12. Select "Spring4.1.6_Lib_New" and Click on "Add External Jars" button.
13. Select the required jar files
commons-logging-1.2.jar

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28**US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

spring-beans-4.1.6.RELEASE.jar
spring-core-4.1.6.RELEASE.jar
spring-context-4.1.6.RELEASE.jar
spring-context-support-4.1.6.RELEASE.jar
spring-expression-4.1.6.RELEASE.jar

14. Click on "Open" Button.
15. Click on "OK" button.
16. Click on "Finish" button.
17. Click on "Apply" button and "OK" button.

Note:

Prepare User Defined Library[Spring4.1.6_Lib_New] only one time and use the same for all Spring applications, not required to prepare for each and every application.

3. Prepare Bean Class

Bean is Reusable Component, it is a normal java Class having Properties and the respective setXXX() and getXXX() methods.

The main intention of Bean classes in Spring applications is to manage properties and their setXXX() and getXXX() methods and some other Business Methods.

To prepare Bean classes in Spring applications we have to use the following Guidelines.

- a) Bean classes must be POJO classes, they must not extend or implement any predefined Library except java.io.Serializable marker interface.
- b) Bean must be declared as "public", "Non-abstract" and "non-final".
---->The main intention of declaring bean class as "public" is to make available bean class scope to IOC Container inorder to create objects.
---->The main intention to declare bean class as "Non-abstract" is to allow to create object .
---->The main intention to declare bean classes as "Non-final" is to extend one bean class to another bean class inorder to improve reusability.
- c) In Bean classes, we have to declare all properties as "private" and all behaviours as "public", it will improve "Encapsulation".

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- d) If we want to provide any constructor in bean class then provide a constructor , it must be 0-arg constructor and "public" constructor, because, IOC Container will search and execute public and 0-arg constructor while instantiating bean.

To create bean class in Eclipse IDE, we have to use the following steps.

1. Right Click on "src" folder.
2. Select "New".
3. Select "Class".
4. Provide package name and class name.

package name: com.durgasoft.beans
class name: HelloBean

5. Click on "Finish" button.
6. As per the requirement provide properties and setXXX() and getXXX() with the following steps.

- a) Declare variables manually.
- b) Right Click and Select "Source".
- c) Select "Generate Getters and Setters".
- d) select "Select All" button.
- e) Click on "OK" button.

7. Provide Business Methods as per the requirement.

EX: HelloBean.java

```
-----  
package com.durgasoft.beans;  
public class HelloBean {  
    public String sayHello(){  
        return "Hello User!";  
    }  
}
```

4 Prepare Bean Configuration File

The main intention of Bean Configuration File is to provide all the bean components configuration details like logical name, fully qualified names of the bean

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

classes, bean classes properties and bean classes dependencies,..... to the IOCcontainer in order to create bean objects and their dependent objects.

Bean Configuration File is an XML file, it will use the following XML tags to configure bean classes.

Spring Framework is able to allow any name to the Configuration file ,but the suggestible name is " applicationContext.xml ".

```
<beans ----XSD---- >  
----  
<bean id="--" class="--">  
----  
</bean>  
----  
</beans>
```

Where "<beans>" tag is a root tag, it will include no of beans configurations.

Where "<bean>" tag can be used to configure single bean class.

Where "id" attribute will take identity name or logical name to the Bean component.

Where "name" attribute will take fully qualified name of the bean class.

To prepare bean configuration file in Eclipse IDE we have to use the following steps.

1) Create a package under "src" folder.

- a) Right Click on "src".
- b) Select "New".
- c) select "package".
- d) Provide package name "com.durgasoft.cfgs".
- e) Click on "Finish" button.

2) Create XML file under package.

- a) Right Click on "com.durgasoft.cfgs" package.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- b) Select "New".
- c) Select "Others".
- d) Select XML and "XML File".
- e) Click on "Next" button.
- f) Provide file name "spring_beans_config.xml".
- g) Click on "Next" button.
- h) Click on "Next" button.
- i) Click on "Finish" button.

3) Provide spring XSD in configuration File.

Open bean.html file available at the location
"E:\softwares\spring\spring-framework-4.1.6.RELEASE\docs\spring-framework-reference\html" copy XSD from any example and past into XML file.

Provide beans configurations as per the requirement.

EX:spring_beans_config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans --- XSD --->
  <bean id="hello" class="com.durgasoft.beans.HelloBean"/>
</beans>
```

5. Prepare Test / Client Appl:

The main intention of Test / Client application is to activate IOC Container , to create Bean Components and to access business methods.

To prepare Test application in Eclipse IDE we have to use the following steps.

- a) Right Click on "SRC".
- b) Select "New".
- c) Select "Class".
- d) Provide the following details.

package name: com.durgasoft.test

Class Name : Test

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- Select "public static void main(String[] args)
e) Click on "Finish" button.

Provide Application logic in main() method with the following steps.

- 1) Activate ApplicationContext IOC Container.
- 2) Get Bean Object from ApplicationContext.
- 3) Access Business Method.

EX: Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.HelloBean;
public class Test {
    public static void main(String[] args)throws Exception {
        ApplicationContext context=new ClassPathXmlApplicationContext
        ("/com/durgasoft/cfgs/spring_beans_config.xml");
        HelloBean bean=(HelloBean) context.getBean("hello");
        System.out.println(bean.sayHello());
    }
}
```

If we provide properties and their respective
setXXX() and getXXX() methods in bean classes then we have to send values to bean
properties in the following two ways.

- 1) From Test Program
- 2) From Beans configuration File.

1) From Test Program

In this approach, we have to set value to bean properties programmatically from Test
application.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**HelloBean.java**

```
package com.durgasoft.beans;

public class HelloBean {
    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String sayHello() {
        return "Hello "+name;
    }
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans.xsd">
    <bean name="helloBean" class="com.durgasoft.beans.HelloBean">
    </bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
```

CONTACT US:**Mobile: +91- 8885 25 26 27****+91- 7207 21 24 27/28****US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.HelloBean;

public class Test {

    public static void main(String[] args) throws Exception{
        ApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
        HelloBean bean=(HelloBean)context.getBean("helloBean");
        bean.setName("Durga");
        System.out.println(bean.sayHello());
    }
}
```

OP: Hello Durga!

In the above approach, we have to recompile the Test application when we change messages, it is not suggestible in application development. To overcome this problem we have to use beans configuration file to prepare messages inorder to send to Bean object.

2. Beans Configuration File:

To provide messages to the bean properties through their setXXX() methods from bean configuration file we have to use the following tag in beans configuration file.

```
<beans ---XSD---->
---
<bean id="--" class="--">
    <property name="--" value="--"/>
---
</bean>
---
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

</beans>

Where <property> tag is able to represent single bean property.

Where "name" attribute in <property> tag is able to take property name to which we want to send data.

Where "value" attribute in <property> tag is able to take the value which we want to send to the bean property.

app2:

HelloBean.java-----
package com.durgasoft.beans;public class HelloBean {
 private String name; public String getName() {
 return name;
 } public void setName(String name) {
 this.name = name;
 } public String sayHello() {
 return "Hello "+name;
 } }

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans.xsd">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean name="helloBean" class="com.durgasoft.beans.HelloBean">
    <property name="name" value="Durga"/>
</bean>
</beans>
```

Test.java

```
-----
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
```

```
import com.durgasoft.beans.HelloBean;
```

```
public class Test {
```

```
    public static void main(String[] args) throws Exception{
        ApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
        HelloBean bean=(HelloBean)context.getBean("helloBean");
        System.out.println(bean.sayHello());
    }
}
```

OP: Hello Durga

Steps to prepare Spring Application[Core Module] in Netbeans IDE:

1. Install Netbeans IDE
2. Create Java Project:
3. Add Spring library to Project Library
4. Prepare Packages under "Source Packages"
5. Prepare Bean classes as per the requirement
6. Prepare Configuration File
7. Create Test class.
8. Run Test class

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. Installation Process:

1. Download netbeans-8.2-windows.exe file from internet.
2. Double click on netbeans-8.2-windows.exe
3. Click on "Yes" button.
4. Click on "Next" button.
5. Select "Checkbox" like "I Accept the terms in licence aggrement".
6. Click on "Next" button.
7. Change Netbeans installation location from C:\Program Files\NetBeans 8.2 to C:\NetBeans 8.2
8. Click on "Next" button.
9. Change Glassfish server installation location from "C:\Program Files \glassfish-4.1.1" to "C:\glassfish-4.1.1" .
10. Click on "Next" button.
11. Click on "Install" button.
12. Click on "Finish".
13. Double Click on "Netbeans" IDE icon on our desktop.

2. Create Java Project:

- a) Right Click on "Projects" Browser.
- b) Select "New Project".
- c) Select "Java" under "Categories".
- d) Select "Java Application" under projects.
- e) Click on "Next" button.
- f) Change application[app21]
- g) Change Project Location[D:\spring3\core\netbeans].
- h) Provide Main class Name[com.durgasoft.test.Test].
- i) Click on "Finish" button.

3. Add Spring library to Project Library:

- a) Right Click on "Libraries" folder in Project.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- b) Select "Add Library".
- c) Select "Spring Framework 4.0.1" Library.
- d) Click on "Add Library" button.

4. Prepare Packages under "Source Packages":

- a) Right Click on "Source Package".
- b) Select "New".
- c) Select "Java Package".
- d) Provide package Name[com.durgasoft.beans].
- e) Click on "Finish" button.

Note: Similarly prepare package com.durgasoft.cfgs and com.durgasoft.test.

4. Prepare Bean classes as per the requirement.

- a) Right Click on "com.durgasoft.beans" package.
- b) Select "New"
- c) Select "Java Class".
- d) Provide class Name[User].
- e) Click on "Finish".
- f) In the generated class declare variables.
- g) Generate setXXX() and getXXX() methods .
 - 1) Right Click .
 - 2) Select "Insert Code".
 - 3) Select "Getters and Setters".
 - 4) Select "User" check box.
 - 5) Click on "Generate".
- h) Provide other methods also as per the requirement.

User.java

```
package com.durgasoft.beans;
public class User {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private String uname;
private String uqual;
private String uage;
private String uaddr;
private String uemail;
private String umobile;

    setXXX()
    getXXX()

public void display_User_Details(){
    System.out.println("User Details");
    System.out.println("-----");
    System.out.println("User Name      :" +uname);
    System.out.println("User Qualification : " +uqual);
    System.out.println("User Age       :" +uage);
    System.out.println("User Address   :" +uaddr);
    System.out.println("User Email     :" +uemail);
    System.out.println("User Mobile    :" +umobile);
}
```

6) Prepare Configuration File:

- a) Right Click on "com.durgasoft.cfgs" under Source Packages.
- b) Click on "New".
- c) Select "Other".
- d) Select "Other" under Categories.
- e) Select "SpringXMLConfig(-)" under Types.
- f) Click on "Next" button.
- g) Provide file name[spring_beans_config.xml].
- h) Click on "Next" button.
- i) Select required "Namespaces" .
- j) Click on "Finish" button.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

k) Provide beans configuration in the generated xml file.**spring_beans_config.xml**

```
<beans>
  <bean id="user" class="com.durgasoft.beans.User">
    <property name="uname" value="Durga"/>
    <property name="uqual" value="MTech"/>
    <property name="uage" value="28"/>
    <property name="uaddr" value="Hyd"/>
    <property name="uemail" value="durga@durgasoft.com"/>
    <property name="umobile" value="91-9988776655"/>
  </bean>
</beans>
```

7) Create Test class:

Test class already created at the time of creating project, where provide application logic.

Test.java

```
-----
package com.durgasoft.test;
import com.durgasoft.beans.User;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class Test {
  public static void main(String[] args)throws Exception {
    ApplicationContext context=new
    ClassPathXmlApplicationContext("/com/durgasoft/cfgs/spring_beans_config.xml");
    User user=(User)context.getBean("user");
    user.display_User_Details();
  }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

IOC CONTAINERS

The main intention of IOC Container is to read bean configurations from configuration file , creating Bean objects and Providing bean Objects to Spring applications.

There are two types of IOC Containers in SPring framework.

1. BeanFactory | it is deprecated for spring 3.x varision.
2. ApplicationContext

1. BeanFactory

- >It is the fundamental or Base container provided by SPring Framework inorder to manage bean objects.
- >BeanFactory IOC container will provide basic functionalities to the spring framework by creating maintaining beans objects as per the beans configuration details which we provided in spring beans configuration file.
- >To represent BeanFactory IOC Container, SPring framework has provided an interface in the form of "org.springframework.beans.factory.BeanFactory".
- >For BeanFactory interface, Spring Framework has provided an implementation class in the form of org.springframework.beans.factory.xml.XmlBeanFactory
- >If we want to use BeanFactory IOC Container in Spring applications then we have to use the following steps.
 - 1) Create Resource Object.
 - 2) Create BeanFactory object
 - 3) Get Bean and access Business Method.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Resource:

--> Resource is an object in SpringFramework, it able to represent all bean configuration details which we provided in beans configurations details.

--> To represent Resource object, Spring Framework has provided a predefined interface in the form of

"org.springframework.core.io.Resource"

--> For Resource interface, Spring Framework has provided the following implementation classes.

1. org.springframework.core.io.ByteArrayResource:

--> It able to represent all the beans configuration details which are available in the form of byte[].

2. org.springframework.core.io.FileSystemResource:

--> It able to get all the beans configuration details which are available in the form of a file in our system

hard disk.

3. org.springframework.core.io.ClassPathResource:

--> It able to get all the beans configuration details which are existed at "classpath" environment variable

refered locations.

4. org.springframework.core.io.InputStreamResource:

--> It able to get all the beans configuration which are existed in the form of InputStream.

5. org.springframework.core.io.UrlResource:

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

--> It able to get all the beans configuration details which are existed at a particular URL in the network.

6. `org.springframework.web.context.support.ServletContextResource`:

--> It able to get all the beans configuration details which are existed in ServletContext.
--> It will be used in spring web applications.

7. `org.springframework.web.portlet.context.PortletContextResource`:

--> It able to get all the beans configuration details which are existed in PortletContext.
--> It will be used in spring web applications designed on the basis of portlets.

EX:

```
Resource res=new ClassPathResource("beans.xml");
```

2) Create BeanFactory Object:

To create XmlBeanFactory class object we have to use the following constructor.

```
public XmlBeanFactory(Resource res)
```

EX: BeanFactory factory=new XmlBeanFactory(res);

3) Get Bean object from BeanFactory and access business method:

To get Bean object from BeanFactory we have to use the following method.

```
public Object getBean(String id_Name)
```

EX:

```
HelloBean bean=(HelloBean)factory.getBean("hello");
```

Note: BeanFactory is deprecated in Spring3.x version.

Example:

`HelloBean.java`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
package com.durgasoft.beans;
```

```
public class HelloBean {  
    static {  
        System.out.println("Bean Loading.....");  
    }  
    public HelloBean() {  
        System.out.println("Bean Created....");  
    }  
    public String sayHello() {  
        return "Hello User";  
    }  
}
```

```
applicationContext.xml
```

```
<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
       xsi:schemaLocation="http://www.springframework.org/schema/beans  
                           http://www.springframework.org/schema/beans/spring-beans.xsd">  
    <bean id="helloBean" class="com.durgasoft.beans.HelloBean"/>  
</beans>
```

```
Test.java
```

```
package com.durgasoft.test;  
  
import org.springframework.beans.factory.xml.XmlBeanFactory;  
import org.springframework.core.io.ClassPathResource;  
  
import com.durgasoft.beans.HelloBean;  
  
public class Test {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public static void main(String[] args) throws Exception{
```

```
    XmlBeanFactory factory = new XmlBeanFactory(new  
ClassPathResource("applicationContext.xml"));  
    HelloBean bean = (HelloBean)factory.getBean("helloBean");  
    System.out.println(bean.sayHello());  
}
```

}

In the above application, when we activate BeanFactory container then BeanFactory Container will be started and it will not create any Bean object immediately and it will perform the following actions.

1. It will take bean configuration file name and location from Resource object.
2. It will search for the respective bean configuration file at the specified location.
3. If the respective bean configuration file is available then Beanfactory container will load that xml file to the memory.
4. After XML file loading, BeanFactory Container will parse that xml file, that is, it will check all the tags in XML file are provided properly or not, all attributes are available properly or not.
5. After the XML file parsing, BeanFactory Container will read data from Beans configuration file and stores in Resource object.

After getting BeanFactory Object, when we access getBean(-) method then BeanFactory will perform the following actions.

1. BeanFactory will search for the respective Bean configuration in Resource object on the basis of the provided identity.
2. If any bean configuration is identified in Resource object on the basis of the provided identity then BeanFactory container will take the respective bean class name and its location.
3. BeanFactory Container will search for the respective bean class at the specified location, if it is available then BeanFactory Container will load all the bean class bytecode to the memory.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4. BeanFactory Container will create Object for the loaded bean class and its dependent bean objects.

5. BeanFactory Container will store the generated bean object and its dependent objects in Container object in the form of Key-Value pairs, where keys must be the "id" attribute values which we specified in beans configuration file and values are Bean Objects.

BeanFactory is following lazy-bean initialization that is bean object is created when we call the getbean method .

ApplicationContext: not at the time of BeanFactory object creation time.

ApplicationContext IOC Container is an extension of **BeanFactory IOC Container**, it able to provide some advanced features like Internationalization, Event Handling,.... along with fundamental functionalities what BeanFactory is providing.

In Spring, ApplicationContext IOC Container is represented in the form of the following predefined interface.

"**org.springframework.context.ApplicationContext**".

SpringFramework has provided ApplicationContext as a child interface to BeanFactory interface.

SpringFramework has provided the following three implementation classes for ApplicationContext.

1. ClassPathXmlApplicationContext:

--> It able to get all the beans configuration details from configuration file which is existed in application classpath.

2. FileSystemXmlApplicationContext:

--> It able to get all the beans configuration details from Configuration file which is existed at our system harddisk.

3. WebXmlApplicationContext:

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

--> It able to get all the beans configuration details from configuration file which is existed in web application.

Example:**HelloBean.java**

```
-----  
package com.durgasoft.beans;  
  
public class HelloBean {  
    static {  
        System.out.println("Bean Loading.....");  
    }  
    public HelloBean() {  
        System.out.println("Bean Created....");  
    }  
    public String sayHello() {  
        return "Hello User";  
    }  
}
```

applicationContext.xml

```
-----  
<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
       xsi:schemaLocation="http://www.springframework.org/schema/beans  
                           http://www.springframework.org/schema/beans/spring-beans.xsd">  
    <bean id="helloBean" class="com.durgasoft.beans.HelloBean"/>  
</beans>
```

Test.java

```
package com.durgasoft.test;  
import org.springframework.beans.factory.xml.XmlBeanFactory;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.core.io.ClassPathResource;  
  
import com.durgasoft.beans.HelloBean;  
  
public class Test {  
  
    public static void main(String[] args) throws Exception{  
  
        ApplicationContext context = new  
ClasspathXmlApplicationContext("applicationContext.xml");  
        HelloBean bean = (HelloBean)context.getBean("helloBean");  
        System.out.println(bean.sayHello());  
    }  
}
```

In the above application, when we activate BeanFactory container then BeanFactory Container will perform the following actions.

1. It will take bean configuration file name and location from Container class constructor.
2. It will search for the respective bean configuration file at the specified location.
3. If the respective bean configuration file is available then ApplicationCntext container will load that xml file to the memory.
4. After XML file loading, ApplicationContext Container will parse that xml file, that is, it will check all the tags in XML file are provided properly or not, all attributes are available properly or not.
5. After the XML file parsing, ApplicationContext Container will read data from Beans configuration file.
6. If any bean configuration is identified in beans configuration file the ApplicationContext container will take beans classes and their locations.
7. ApplicationContext Container will search for the respective beans at the specified locations, if they are available then IOC Container will load all the bean classes bytecode to the memory.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

8. ApplicationContext Container will create Objects for the loaded bean classes and their dependent bean objects.

9. ApplicationContext Container will store all the bean objects and their dependent objects in Container object in the form of Key-Value pairs, where keys must be the "id" attribute values which we specified in beans configuration file.

In the above context, if we access `getBean("--")` method over Container reference then ApplicationContext Container will search for the Bean object on the basis of the provided bean identity , if it is available then ApplicationContext Container will return Bean object.

Q) What are the differences between BeanFactory and ApplicationContext IOC Containers?

Ans:

1) BeanFactory is fundamental IOC Container , it able to provide fundamental functionalities to the spring applications like creating and maintaining bean objects.

ApplicationContext IOC Container is an extension of BeanFactory IOC Container , it able to provide some advanced features like Internationalization, Event Handling,..... along with fundamental functionalities what BeanFactory is providing.

2) BeanFactory is not supporting to integrate AOP services like Security, JTA,... to the spring applications.

ApplicationContext is supporting to integrate AOP services like Security, JTA,... to the spring applications.

3) BeanFactory is not suitable for web applications which we are going to prepare on the basis of Spring web module.

ApplicationContext is suitable for the web applications which we want to prepare on the basis of Spring web module.

4) BeanFactory is able to prepare Singleton objects when we send first request for bean, that is, Lazy Instantiation/Initialization.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

ApplicationContext is able to prepare Singleton objects when we activate Container , that is , early Instantiation/Initialization.

5.BeanFactory is supporting only the scopes like Singleton and Prototype.

ApplicationContext is supporting almost all the Spring scopes like Singleton, Prototype, request, session, globalSession, webSocket,...

6.BeanFactory is mainly for Standalone Applications.

ApplicationContext is for all the types of Spring framework applications.

7.BeanFactory is an outdated Container in Spring applications.

ApplicationContext is not outdated Container.

Beans In Spring Framework

Bean Definition: Bean is a Software Reusable Component, it is a normal java class contains properties and the corresponding setXXX(-) and getXXX() methods and which are created and managed by IOC Container in Spring Framework.

Rules and Regulations to write Bean classes:

a) Bean classes must be POJO classes, they must not extend or implement any predefined Library except java.io.Serializable marker interface.

b) Bean must be declared as "public" , "Non-abstract" and "non-final".

----> The main intention of declaring bean class as "public" is to make available bean class scope to IOC Container inorder to create objects.

----> The main intention to declare bean class as "Non-abstract" is to allow to create object .

----> The main intention to declare bean classes as "Non-final" is to extend one bean class to another bean class inorder to improve reusability.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- c) In Bean classes, we have to declare all properties as "private" and all behaviours as "public", it will improve "Encapsulation".
- d) If we want to provide any constructor in bean class then provide a constructor , it must be 0-arg constructor and "public" constructor, because, IOC Container will search and execute public and 0-arg constructor while instantiating bean.

If we want to use Beans in Spring applications then we must configure that bean classes in spring beans configuration file , because, IOCContainer will recognize and create Bean objects by getting bean class details from beans configuration file only.

There are three ways to provide beans configurations in spring applications.

1. XML Configuration
2. Java Based Configuration
3. Annotations Configuration

1. XML Configuration

To provide beans configurations in beans configuration file we have to use the following xml tags.

```
<beans>
  <bean id="--" name="--" class="--" scope="--">
    <property name="--" value="--"/>
  </bean>
</beans>
```

Where **<beans>** tag is root tag in beans configuration file.

Where **<bean>** tag is able to provide configuration details of a particular bean class.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where "class" attribute in <bean> is able to provide fully qualified name of the bean class.

Where <property> attribute is able to represent a particular property[variable] in bean class and it will set the specified value to the respective bean property by executing setXXX(-) method.

Q) What is the difference between "id" attribute and "name" attribute in <bean> tag?

Ans: 'id' attribute is able to take exactly one identity to the bean object, it will not allow more than one identity.

'name' attribute in <bean> tag is able to allow more than one identity name to the bean object, where in multiple values only first value is treated as the actual bean identity and the remaining names are alias names for the bean object. In this context, while providing alias names to the bean object we have to use either ',' or ';' or [space] as delimiter[seperator].

EX1:

```
<beans>
<bean id="bean1" class="com.durgasoft.beans.MyBean"/>
</beans>
MyBean mb=(MyBean)ctx.getBean("bean1");
Status: Valid.
```

EX2:

```
<beans>
<bean id="bean1 bean2 bean3" class="com.durgasoft.beans.MyBean"/>
</beans>
MyBean mb=(MyBean)ctx.getBean("bean1");
Status: org.springframework.beans.factory.NoSuchBeanDefinitionException: No bean named 'bean1' is defined
```

Note: Similarly the following cases are also be invalid.

```
<bean id="bean1,bean2,bean3" class="MyBean"/>->INvalid
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean id="bean1;bean2;bean3" class="MyBean"/>->Invalid  
<bean id="bean1bean2bean3" class="MyBean"/>-> Valid
```

EX3:

```
<beans>  
  <bean name="bean1" class="MyBean"/>  
</beans>  
MyBean mb=(MyBean)ctx.getBean("bean1");  
Status: Valid
```

EX4:

```
<bean name="bean1 bean2 bean3" class="MyBean"/>  
MyBean mb1=ctx.getBean("bean1");  
MyBean mb2=ctx.getBean("bean2");  
MyBean mb3=ctx.getBean("bean3");  
Status: Valid  
Similarly the following cases are also be valid.  
<bean name="bean1,bean2,bean3" class="MyBean"/>  
<bean name="bean1;bean2;bean3" class="MyBean"/>
```

In bean Configuration,we can provide bean class configuration with 'name' attribute and without id attribute.

Note: It is possible to use both 'id' attribute and 'name' attribute in single <bean> tag.

where id attribute value is treated as bean identity and 'name' attribute values are treated as bean alias names.

EX5:

```
<bean id="bean1" name="bean2" class="MyBean"/>  
MyBean mb1=ctx.getBean("bean1");  
MyBean mb2=ctx.getBean("bean2");  
Status: Valid.
```

Note: It is possible to provide bean alias names explicitly from out side of the bean definition in configuration file by using <alias> tag.

```
<alias name="--" alias="--"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where "name" attribute will take bean logical name which we specified with "id" attribute in beans configuration file.

Where "alias" attribute will take alias name.

EX6:

```
<beans>
<bean name="bean1" class="MyBean"/>
<alias name="bean1" alias="bean2"/>
<alias name="bean2" alias="bean3"/>
</bean>
</beans>
```

MyBean mb1=ctx.getBean(bean1); --> Valid

MyBean mb2=ctx.getBean(bean2); --> Valid

MyBean mb3=ctx.getBean(bean3); --> Valid

Bean Scopes:

In J2SE applications, we are able to define scopes to the data by using the access modifiers like public, protected, <default> and private.

Similarly, in Spring framework to define scopes to the beans spring framework has provided the following scopes.

- 1. singleton Scope [Default Scope]
 - 2. prototype Scope
 - 3. request Scope
 - 4. session Scope
 - 5. globalSession Scope
 - 6. application Scope
 - 7. webSocket scope
- we are used in core module
- these scope we are used in web Module or web mvc model

1. singleton Scope:

It is default scope in Spring applications.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

If we use this scope to the bean then IOCContainer will create Single Bean object for single bean definition in Spring config file.

This approach will return the same bean object for every time requesting bean object. When we request bean object first time then IOCContainer will create bean object really and it will be stored in Cache memory, then , every time accessing bean object , IOCContainer will return the same bean object reference value with out creating new Bean objects.

EX:

beans.xml

```
<beans>
  <bean id="bean1" class="com.durgasoft.MyBean" scope="singleton"/>
  <bean id="bean2" class="com.durgasoft.MyBean" scope="singleton"/>
</beans>
```

```
System.out.println(ctx.getBean("bean1")); //MyBean@a111
```

```
System.out.println(ctx.getBean("bean1")); //MyBean@a111
```

```
System.out.println(ctx.getBean("bean2")); //MyBean@a222
```

```
System.out.println(ctx.getBean("bean2")); //MyBean@a222
```

2. prototype Scope:

It is not default Scope in Spring framework.

In Spring applications, if we provide "prototype" scope in bean configuration file then IOCContainer will create a new Bean object at each and every time of calling getBean(--) method.

EX:

beans.xml

```
<beans>
```

```
  <bean id="bean1" class="com.durgasoft.MyBean" scope="prototype"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean id="bean2" class="com.durgasoft.MyBean" scope="prototype"/>
</beans>
```

```
System.out.println(ctx.getBean("bean1")); //MyBean@a111
System.out.println(ctx.getBean("bean1")); //MyBean@a222
System.out.println(ctx.getBean("bean2")); //MyBean@a333
System.out.println(ctx.getBean("bean2")); //MyBean@a444
```

3. requestScope:

This scope is not usefull in Standalone Applications[Spring Core MOdule], it will be used in Web applications which are prepared on the basis of Spring Web module.

RequestScope is able to create a seperate bean object for each and every request object.

4. sessionScope:

This Scope will be used web applications which are prepared on the basis of Spring web module and it is not applicable in Standalone Applications.

sessionScope allows to create a seperate bean object for each and every Session object in web applications.

5. globalSession Scope:

This scope is not usefull in standard applications, it is usefull in portlet applications which are prepared on the basis of SPring web module.

globalSession scope allows to create a seperate bean object for each and every portlet Session.

6. application Scope:

This scope is not usefull in standalone Applications, it is usefull in web applications prepared on the basis of Spring web momdule.

ApplicationScope allows to create a seperate bean object for each and every ServletContext object.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

7. webSocketScope:

This scope is usefull in web applications which are prepared on the basis of spring web module.

websocket scope allows to create a seperate bean object for single websocket lifecycle.

If we use the scopes like request, session, globalSession, application, webSocket,... in standalone applications which are prepared on the basis of spring core module then Container will rise an exception like "java.lang.IllegalStateException".

Note: Spring Framework has provided environment to customize the existed scopes , but, it is not suggestible. Spring framework has provided environment to create new scopes in spring applications.

To define and use custom scopes in Spring Framework we have to use the following steps.

- 1.Create User Defined Scope class.
- 2.Register User defined Scope in Spring beans configuration file.
- 3.Use User defined Scope to the Beans in beans configuration file.

1. Create User defined Scope class:

- a) Declare an user defined class.
- b) Implement org.springframework.beans.factory.config.Scope interface to User defined class.

c) Implement the following Scope interface methods in User defined class.

1) get(): It able to generate a bean object from Scope.

```
public Object get(String name, ObjectFactory factory)
```

2) remove(): It able to remove bean object from Scope.

```
public Object remove(String name)
```

3) getConversationId(): It able to provide an id value of the scope if any.

EX: IN Session scope, generating sessionId.

```
public String getConversationId()
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4) `registerDestructionCallback()`: It will be executed when bean object is destroyed in Scope

```
public void registerDestructionCallback(String name, Runnable r)
```

5) `resolveContextualObject()`: It will resolve the situation where Multiple Context objects associated with the keys.

```
public Object resolveContextualObject(String name)
```

Note: From the above methods, `get()` and `remove()` methods are mandatory to implement and all the remaining methods are optional.

2. Register User defined Scope in Spring beans configuration File:

a) Configure `org.springframework.beans.factory.config.CustomScopeConfigurer` class as a bean.

b) Declare "scopes" as property in `CustomScopeConfigurer`

c) Declare "map" under scopes property.

d) Declare "entry" under the "map".

e) Declare "key" in "entry" with User defined scope name and provide value as User defined SCope object.

3. Apply User defined Scope to bean definitions:

Use "scope" attribute in `<bean>` tag to apply user defined scope.

Note: In the following example, we have defined `threadScope` as user defined scope, that is, it able to create a separate bean object for each and every thread.

Example:

`CustomThreadLocal.java`

```
package com.durgasoft.scopes;
import java.util.HashMap;
```

```
public class CustomThreadLocal extends ThreadLocal<Object> {
    @Override
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
protected Object initialValue() {
    return new HashMap<String, Object>();
}
}
```

ThreadScope.java

```
package com.durgasoft.scopes;

import java.util.Map;

import org.springframework.beans.factory.ObjectFactory;
import org.springframework.beans.factory.config.Scope;

public class ThreadScope implements Scope {

    Map<String, Object> scope = null;
    CustomThreadLocal threadLocal = new CustomThreadLocal();
    @Override
    public Object get(String name, ObjectFactory objectFactory) {

        scope = (Map<String, Object>)threadLocal.get();
        Object obj = scope.get(name);
        if(obj == null) {
            obj = objectFactory.getObject();
            scope.put(name, obj);
        }
        return obj;
    }

    @Override
    public String getConversationId() {
        // TODO Auto-generated method stub
        return null;
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

```
@Override  
public void registerDestructionCallback(String arg0, Runnable arg1) {  
    // TODO Auto-generated method stub
```

}

```
@Override  
public Object remove(String name) {  
    Object obj = scope.remove(name);  
    return obj;  
}
```

```
@Override  
public Object resolveContextualObject(String arg0) {  
    // TODO Auto-generated method stub  
    return null;  
}
```

}

HelloBean.java

```
package com.durgasoft.beans;  
  
public class HelloBean {  
    public HelloBean() {  
        System.out.println("HelloBean Object is created");  
    }  
    public String sayHello() {  
        return "Hello User from "+Thread.currentThread().getName()+" Scope";  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
package com.durgasoft.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.HelloBean;
import com.durgasoft.scopes.ThreadScope;

public class Test {

    public static void main(String[] args) {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        HelloBean bean1 = (HelloBean)context.getBean("helloBean");
        HelloBean bean2 = (HelloBean)context.getBean("helloBean");
        System.out.println(bean1);
        System.out.println(bean2);
        System.out.println(bean1 == bean2);
        System.out.println(bean1.sayHello());
        System.out.println(bean2.sayHello());

        ThreadScope threadScope = (ThreadScope)context.getBean("threadScope");
        HelloBean bean3 = (HelloBean)threadScope.remove("helloBean");
        System.out.println(bean3);

        HelloBean bean4 = (HelloBean)context.getBean("helloBean");
        HelloBean bean5 = (HelloBean)context.getBean("helloBean");
        System.out.println(bean4);
        System.out.println(bean5);
        System.out.println(bean4 == bean5);
        System.out.println(bean4.sayHello());
        System.out.println(bean5.sayHello());
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans.xsd">
    <bean id="helloBean" class="com.durgasoft.beans.HelloBean" scope="thread"/>
    <bean id="threadScope" class="com.durgasoft.scopes.ThreadScope"/>
    <bean id="scopeConfigurer"
          class="org.springframework.beans.factory.config.CustomScopeConfigurer">
        <property name="scopes">
            <map>
                <entry key="thread" value-ref="threadScope"/>
            </map>
        </property>
    </bean>
</beans>
```

2. Java Based Configuration

In Spring, upto Spring2.4 version Spring beans configuration file is mandatory to configure bean classes and their metadata, but, Right from Spring3.x version Spring beans configuration file is optional, because, SPring3.x version has provided Java Based Configuration as replacement for XML documents.

If we want to use Java Based Configuration as an alternative to Spring beans configuration file in Spring applications then we have to use the following steps.

1. Create Bean classes as per the requirement.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Create Beans configuration class with the following annotations.

`org.springframework.context.annotation.@Configuration`

--> It able to represent a class as configuration class.

`org.springframework.context.annotation.@Bean`

--> It will be used at method to represent the return object is bean object.

3. In Test class, Create ApplicationContext object with the `org.springframework.context.annotation.AnnotationConfigApplicationContext` implementation class.

`ApplicationContext context=new AnnotationConfigApplicationContext(BeanConfig.class);`

4. Get Bean object from ApplicationContext by using the following method.

`public Object getaBean(Class c)`

EX: Bean b=context.getBean(Bean.class);

5. Access business methods from Bean.

Example:

HelloBean.java

```
package com.durgasoft.beans;

public class HelloBean {
    static {
        System.out.println("Bean Loading.....");
    }
    public HelloBean() {
        System.out.println("Bean Created....");
    }
    public String sayHello() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        return "Hello User";
    }
}
```

HelloBeanConfig.java

```
package com.durgasoft.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import com.durgasoft.beans.HelloBean;

@Configuration
public class HelloBeanConfig {
    @Bean
    public HelloBean helloBean(){
        return new HelloBean();
    }
}
```

Test.java

```
package com.durgasoft.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import com.durgasoft.beans.HelloBean;
import com.durgasoft.config.HelloBeanConfig;

public class Test {

    public static void main(String[] args) throws Exception{

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
ApplicationContext context = new  
AnnotationConfigApplicationContext(HelloBeanConfig.class);  
HelloBean bean = (HelloBean)context.getBean("helloBean");  
System.out.println(bean.sayHello());  
}  
}
```

Bean Lifecycle:

In spring framework applications, when IOC Container recognizes all the beans definitions in beans configuration file then IOC Container will execute that bean by using the following lifecycle actions.

1. Bean Class Loading
2. Bean Instantiation
3. Bean Initialization
4. Bean Destruction

1. Bean Class Loading:

When IOC Container recognized fully qualified names of the bean classes in beans configuration file then IOC Container will load the specified bean class byte code to the memory. To load bean class bytecode to the memory, IOC Container will use the following method.

```
public static Class forName(String class_Name) throws ClassNotFoundException  
EX: Class c=Class.forName("com.durgasoft.beans.Welcom  
eBean");
```

2. Bean Instantiation

In Spring applications, after loading bean class bytecode to the memory, IOC Container will create object for the bean class.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

IN Spring applications we are able to use the following three approaches to create Bean objects.

1. By using Constructor directly.
2. By Using Static Factory Method
3. By Using Instance Factory Method

1. Bean Instantiation through Constructors:

If we want to create bean objects by using constructors then we must provide 0-arg constructor in bean class irrespective of bean class constructor scopes.

Note: In Bean class , if we provide parameterized constructor then IOC Container will rise an exception.

EX:

HelloBean.java

```
public class HelloBean{  
    public HelloBean(){  
        System.out.println("Bean Instantiation");  
    }  
    public String sayHello(){  
        System.out.println("Hello User!");  
    }  
}
```

applicationContext.xml

```
<beans>  
    <bean id="hello" class="HelloBean"/>  
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
class Test{
public static void main(String[] args){
ApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
HelloBean bean=(HelloBean)context.getBean("hello");
System.out.println(bean.sayaHello());
}
}
```

1. Bean Instantiation through Static Factory Method:

In this approach, first we have to define static factory method in Bean class and we have to configure that static factory method in bean definition in beans configuration file.

EX:**HelloBean.java**

```
public class HelloBean{
public static HelloBean getInstance(){
System.out.println("Static Factory Method");
return new HelloBean();
}
public String sayHello(){
System.out.println("Hello User!");
}
}
```

applicationContext.xml

```
<beans>
<bean id="hello" class="HelloBean"
factory-method="getInstance"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

</beans>

Test.java

```
class Test{
public static void main(String[] args){
ApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
HelloBean bean=(HelloBean)context.getBean("hello");
System.out.println(bean.sayaHello());
}
}
```

2. Bean Instantiation through Instance Factory Method:

In this approach, we have to define a separate factory class with instance factory method and we have to configure factory class as bean in beans configuration file then we have to configure factory class and factory method in the original bean class definition by using "factory-bean" and "factory-method" attributes.

EX:

HelloBeanFactory.java

```
public class HelloBeanFactory{
public HelloBean getBeanInstance(){
SYstem.out.println("Instance Factory Method");
return new HelloBean();
}
```

HelloBean.java

```
public class HelloBean{
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public String sayHello(){  
System.out.println("Hello User!");  
}  
}
```

applicationContext.xml

```
<beans>  
<bean id="hello" class="HelloBean"  
factory-method="getBeanInstance"  
factory-bean="factory" />  
<bean id="factory" class="HelloBeanFactory"/>  
</beans>
```

Test.java

```
class Test{  
public static void main(String[] args){  
ApplicationContext context=new  
ClassPathXmlApplicationContext("applicationContext.xml");  
HelloBean bean=(HelloBean)context.getBean("hello");  
System.out.println(bean.sayHello());  
}  
}
```

Bean Initialization and Bean Destruction:

As part of Beans lifecycle, IOC Container has to perform Beans initialization after the Bean Instantiation and IOC Container has to perform Bean destruction after executing the business logic or at the time of shutdown the IOC Container.

There are three ways to perform Beans initialization and destruction in Spring Framework.

1. By using Custom initialization and destruction methods.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. By using InitializingBean and DesposableBean callback interfaces.
3. By using @PostConstruct and @Predestroy annotations

1. By using Custom initialization and destruction methods:

In this approach, we have to define user defined initialization and destruction methods with any name and we have to configure that user defined methods in beans definitions in beans configuration file by using "init-method" and "destroy-method" attributes in <bean> tag.

EX:

WelcomeBean.java

```
package com.durgasoft.beans;

public class WelcomeBean {
    public void init(){
        System.out.println("User defined init() method");
    }
    public String sayWelcome(){
        return "Welcome To Durga Software Solutions";
    }
    public void destroy(){
        System.out.println("User defined destroy() Method");
    }
}
```

applicationContext.xml

```
<beans>
    <bean id="wel"      class="com.durgasoft.beans.WelcomeBean"
          init-method="init" destroy-method="destroy" />
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
package com.durgasoft.test;

import org.springframework.context.support.AbstractApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.WelcomeBean;
public class Test {
public static void main(String[] args) throws Exception {
AbstractApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
WelcomeBean bean=(WelcomeBean)context.getBean("wel");
System.out.println(bean.sayWelcome());
context.registerShutdownHook();
}
}
```

2. By using InitializingBean and DisposableBean callback interfaces:

IN Spring framework, **InitializingBean** is a callback interface , it provides the following method to execute while performing bean initialization by IOC Container.

```
public void afterPropertiesSet() throws Exception
```

Note: This method will be executed by the container after executing all the setXXX() methods of bean class by ApplicationContext.

IN Spring framework, **DisposableBean** is a callback interface, it provides the following method to execute while performing Bean Destruction by IOC Container.

```
public void destroy()
```

EX:

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

WelcomeBean.java

```
package com.durgasoft.beans;
import org.springframework.beans.factory.DisposableBean;
import org.springframework.beans.factory.InitializingBean;

public class WelcomeBean implements InitializingBean,DisposableBean{
private String message;
public String getMessage() {
return message;
}
public void setMessage(String message) {
this.message = message;
System.out.println("setMessage()");
}
public String sayWelcome(){
return message;
}
@Override
public void afterPropertiesSet() throws Exception {
message="Welcome To Durga Software Solutions";
System.out.println("afterPropertiesSet()");
}
@Override
public void destroy() throws Exception {
System.out.println("destroy()");
}
}
```

applicationContext.xml

```
<beans>
<bean id="wel" class="com.durgasoft.beans.WelcomeBean">
<property name="message" value="Welcome To Durgasoft"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.support.AbstractApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.WelcomeBean;
public class Test {
public static void main(String[] args) throws Exception {
AbstractApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
WelcomeBean bean = (WelcomeBean) context.getBean("wel");
System.out.println(bean.sayWelcome());
context.registerShutdownHook();
}
}
```

3. By using **@PostConstruct** and **@Predestroy** annotations

@PostConstruct annotation will make a method to execute by the IOC Container while performing Beans initialization.

@Predestroy annotation will make a method to execute by the IOC Container while performing Bean Destruction.

EX:

Welcome.java

```
package com.durgasoft.beans;

import javax.annotation.PostConstruct;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import javax.annotation.PreDestroy;

public class WelcomeBean {
    private String message;

    public String getMessage() {
        return message;
    }

    public void setMessage(String message) {
        this.message = message;
        System.out.println("setMessage()");
    }

    public String sayWelcome(){
        return message;
    }

    @PostConstruct
    public void init(){
        System.out.println("postConstruct method");
    }

    @PreDestroy
    public void destroy(){
        System.out.println("preDestroy method");
    }
}

applicationContext.xml

<beans>
    <context:annotation-config/>
    <bean id="wel" class="com.durgasoft.beans.WelcomeBean">
        <property name="message" value="Welcome To Durgasoft"/>
    </bean>
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: We need to provide "context" name space along with beans name space in spring beans configuration file.

Test.java

```
public class Test {  
    public static void main(String[] args) throws Exception {  
        AbstractApplicationContext context = new  
        ClassPathXmlApplicationContext("applicationContext.xml");  
        WelcomeBean bean = (WelcomeBean) context.getBean("wel");  
        System.out.println(bean.sayWelcome());  
        context.registerShutdownHook();  
    }  
}
```

To destroy the object explicitly we have to call this method to distract the object from container we must used these method. this method is available in AbstractApplicationContext interface it is child interface of ApplicationContext.

Note: IN general, in spring framework applications, we are able to use either of the above three approaches to perform beans initialization and destruction, we are not using all three approaches at a time. If we use all the above three approaches in single bean then IOC Container will execute the above three approaches provided intialization methods and destruction methods in the following order.

Initialization order:

- a) an initialization method marked with @Postconstruct annotation.
- b) afterPropertiesSet() method provided by InnitializingBean callback interface.
- c) an initialization method configured with "init-method" in <bean> tag in beans configuration file

Destruction Order:

- a) A Destruction method marked with @Predestroy annotation
- b) destroy() method provided by DisposableBean callback interface.
- c) A destruction method configured with "destroy-method" annotation in <bean> tag in beans configuration file.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: If we have same initialization method and destruction method in more than one bean class in spring application then we are able to provide common init method and destroy method configuration by using "default-init-method" attribute and "default-destroy-method" attribute in <beans> tag [not in <bean> tag] with out providing "init-method" and "destroy-method" attributes in <bean> tag at each and every bean configuration.

EX:**Welcome.java**

```
public class Welcome {  
    public void init(){  
        System.out.println("init()-Welcome");  
    }  
    public void destroy(){  
        System.out.println("destroy()-Welcome");  
    }  
}
```

Hello.java

```
public class Hello{  
    public void init(){  
        System.out.println("init()-Hello");  
    }  
    public void destroy(){  
        System.out.println("destroy()-Hello");  
    }  
}
```

spring beans config.xml

```
<beans default-init-method="init" default-destroy-method="destroy">  
    <bean id="welcome" class="com.durgasoft.beans.Welcome"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean id="hello" class="com.durgasoft.beans.Hello"/>
</beans>
```

Test.java

```
public class Test{
public static void main(String[] args){
AbstractApplicationContext context=new ClassPathXmlApplicationContext(
"/com/durgasoft/cfgs/spring_beans_config.xml");
Welcome bean1=(Welcome)context.getBean("welcome");
Hello bean2=(Hello)context.getBean("hello");
context.registerShutdownHook();
}
}
```

4. Beans Inheritance:

In general, in Spring framework, we are able to provide more and more no of configuration details in beans definitions under beans configuration file like properties configurations, dependency injection related configurations, initialization destruction methods configurations,.....

In the above context, we are able to reuse one bean configuration details in another bean configurations like normal java classes inheritance inorder to improve reusability and inorder to reduce no of configurations in spring configuration file by declaring parent and child beans configurations.

To declare a particular bean configuration as parent to the present bean configuration then we have to use "parent" attribute in "<bean>" tag in beans configuration file, where "parent" attribute will take identity of the respective bean definition.

EX:

WishBean.java

```
package com.durgasoft.beans;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class WishBean {  
    private String wish_Message;  
    private String name;  
  
    public void init(){  
        System.out.println("WishBean Initialization");  
    }  
    public void destroy(){  
        System.out.println("WishBean Destruction");  
    }  
  
    setXXX()  
    getXXX()  
}
```

HelloBean.java

```
package com.durgasoft.beans;  
public class HelloBean {  
    private String wish_Message;  
    private String name;  
  
    public void init(){  
        System.out.println("HelloBean Initialization");  
    }  
    public void destroy(){  
        System.out.println("HelloBean Destruction");  
    }  
  
    setXXX()  
    getXXX()  
  
    public String sayHello(){  
        return wish_Message+name;  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
}
```

WelcomeBean.java

```
package com.durgasoft.beans;
public class WelcomeBean {
    private String wish_Message;
    private String name;

    public void init(){
        System.out.println("WelcomeBean Initialization");
    }
    public void destroy(){
        System.out.println("WelcomeBean Destruction");
    }

    setXXX()
    getXXX()

    public String sayWelcome(){
        return wish_Message+name;
    }
}
```

spring_beans_config.xml

```
<beans>
    <bean id="wishBean" class="com.durgasoft.beans.WishBean" init-method="init"
destroy-method="destroy">
        <property name="wish_Message" value="Durga Software Solutions"/>
        <property name="name" value="Anil"/>
    </bean>
    <bean id="helloBean" class="com.durgasoft.beans.HelloBean" parent="wishBean">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="wish_Message" value="Hello "/>
</bean>
<bean id="welcomeBean" class="com.durgasoft.beans.WelcomeBean"
parent="wishBean">
    <property name="wish_Message" value="Welcome "/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.support.AbstractApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.HelloBean;
import com.durgasoft.beans.WelcomeBean;
public class Test {
    public static void main(String[] args) throws Exception {
        AbstractApplicationContext context = new
        ClassPathXmlApplicationContext("/com/durgasoft/cfgs/  spring_beans_config.xml");
        HelloBean hello = (HelloBean) context.getBean("helloBean");
        System.out.println(hello.sayHello());
        System.out.println();
        WelcomeBean welcome = (WelcomeBean) context.getBean("welcomeBean");
        System.out.println(welcome.sayWelcome());
        context.registerShutdownHook();
    }
}
```

In the above Beans Inheritance, it is possible to declare parent bean definition as a template, that is, a set of configurations which we want to apply to child definitions, not to have its own bean class and not to apply to its own bean class.

To declare a bean definition as template we have to use "abstract" attribute with "true" value in `<bean>` tag , in this context, it is not required to use "class" attribute in bean tag.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:

```
<beans>
<bean id="wishBean" abstract="true" .... >

</bean>
<bean id="helloBean" class="HelloBean" parent="wishBean">

</bean>
<bean id="welBean" class="WelcomeBean" parent="wishBean">

</bean>
</beans>
```

Example:**WishBean.java**

```
package com.durgasoft.beans;

public class WishBean {
    private String wish_Message;
    private String name;

    public void init(){
        System.out.println("WishBean Initialization");
    }
    public void destroy(){
        System.out.println("WishBean Destruction");
    }
    public String getWish_Message() {
        return wish_Message;
    }
    public void setWish_Message(String wish_Message) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
this.wish_Message = wish_Message;  
}  
public String getName() {  
    return name;  
}  
public void setName(String name) {  
    this.name = name;  
}  
}
```

HelloBean.java

```
package com.durgasoft.beans;  
  
public class HelloBean {  
    private String wish_Message;  
    private String name;  
  
    public void init(){  
        System.out.println("HelloBean Initialization");  
    }  
    public void destroy(){  
        System.out.println("HelloBean Destruction");  
    }  
  
    public String getWish_Message() {  
        return wish_Message;  
    }  
    public void setWish_Message(String wish_Message) {  
        this.wish_Message = wish_Message;  
    }  
    public String getName() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }

    public String sayHello(){
        return wish_Message+name;
    }
}
```

WelcomeBean.java

```
package com.durgasoft.beans;

public class WelcomeBean {
    private String wish_Message;
    private String name;

    public void init(){
        System.out.println("WelcomeBean Initialization");
    }
    public void destroy(){
        System.out.println("WelcomeBean Destruction");
    }

    public String getWish_Message() {
        return wish_Message;
    }
    public void setWish_Message(String wish_Message) {
        this.wish_Message = wish_Message;
    }
    public String getName() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }

    public String sayWelcome(){
        return wish_Message+name;
    }
}
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.support.AbstractApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.HelloBean;
import com.durgasoft.beans.WelcomeBean;
import com.durgasoft.beans.WishBean;
public class Test {
    public static void main(String[] args)throws Exception {
        AbstractApplicationContext context=new
        ClassPathXmlApplicationContext("applicationContext.xml");

        //WishBean wish = (WishBean)context.getBean("wishBean");---> Exception
        //System.out.println(wish);
        HelloBean hello=(HelloBean)context.getBean("helloBean");
        System.out.println(hello.sayHello());
        System.out.println();
        WelcomeBean welcome=(WelcomeBean)context.getBean("welcomeBean");
        System.out.println(welcome.sayWelcome());
        context.registerShutdownHook();
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="wishBean" abstract="true" init-method="init" destroy-method="destroy">
        <property name="wish_Message" value="Durga Software Solutions"/>
        <property name="name" value="Anil"/>
    </bean>
    <bean id="helloBean" class="com.durgasoft.beans.HelloBean" parent="wishBean">
        <property name="wish_Message" value="Hello "/>
    </bean>
    <bean id="welcomeBean" class="com.durgasoft.beans.WelcomeBean"
          parent="wishBean">
        <property name="wish_Message" value="Welcome "/>
    </bean>
</beans>
```

Nested Beans:

Declaring a bean configuration in another bean configuration is called as Inner Beans or Nested Beans.

IN Bean class, if we declare any property of User defined class type then we have to use `<bean>` tag for the user defined class as value to the respective property under `<property>` tag.

```
<beans>
```

```
    <bean .... >
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="variable of USer defined class type">
    <bean id="--" class="--"/>
</property>

</bean>
```

```
</beans>
```

EX:

Account.java

```
package com.durgasoft.beans;
public class Account {
    private String accNo;
    private String accName;
    private String accType;
    private long balance;
    setXXX()
    getXXX()
}
```

Employee.java

```
package com.durgasoft.beans;
public class Employee {
    private String eid;
    private String ename;
    private float esal;
    private String eaddr;
    private Account acc;
    setXXX()
    getXXX()
}
```

```
public void displayEmpDetails(){
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
System.out.println("Employee Details");
System.out.println("-----");
System.out.println("Employee Id    :" + eid);
System.out.println("Employee Name  :" + ename);
System.out.println("Employee Salary :" + esal);
System.out.println("Employee Address:" + eaddr);
System.out.println();
System.out.println("Account Details");
System.out.println("-----");
System.out.println("Account Number :" + acc.getAccNo());
System.out.println("Account Name   :" + acc.getAccName());
System.out.println("Account Type   :" + acc.getAccType());
System.out.println("Account Balance :" + acc.getBalance());
}

}
```

applicationContext.xml

```
<beans>
<bean id="emp" class="com.durgasoft.beans.Employee">
<property name="eid" value="E-111"/>
<property name="ename" value="Durga"/>
<property name="esal" value="10000"/>
<property name="eaddr" value="Hyd"/>
<property name="acc">
<bean id="account" class="com.durgasoft.beans.Account">
<property name="accNo" value="abc123"/>
<property name="accName" value="Durga"/>
<property name="accType" value="Savings"/>
<property name="balance" value="20000"/>
</bean>
</property>
</bean>
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
package com.durgasoft.test;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import com.durgasoft.beans.Employee;  
public class Test {  
    public static void main(String[] args) throws Exception {  
        ApplicationContext context = new  
        ClassPathXmlApplicationContext("applicationContext.xml");  
        Employee emp = (Employee) context.getBean("emp");  
        emp.displayEmpDetails();  
    }  
}
```

BeanPostProcessor:

In Spring Framework, when we activate ApplicationContext container then ApplicationContext container will perform the following actions automatically without having developers interaction.

1. Application Context will read beans definitions from beans configuration file.
2. Application Context will recognize beans classes and creates objects for bean classes
3. Application Context will perform initialization by executing initialization methods.
4. When IOCContainer is shutdown then bean objects are destroyed.

IN the above Bean lifecycle, if we want to provide customizations over beans initializations then we have to use a predefined interface provided by Spring framework like "org.springframework.beans.factory.config.BeanPostProcessor".

BeanPostProcessor contains the following two methods .

1. **public void postProcessBeforeInitialization(Object bean, String name) throws BeansException**

--> This method will be executed just before performing bean initialization and

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

immediately after bean instantiation, that is, before executing init() method if we provide custom initialization method init().

2. public void postProcessAfterInitialization(Object bean, String name) throws BeansException

--> This method will be executed immediately after performing beans initialization, that is, after executing init() method if we provide custom initialization method init().

To use BeanPostProcessor in Spring applications we have to declare an user defined class as an implementation class to BeanPostProcessor interface , we must implement the above specified methods and we must configure implementation class in bean configuration file. If we provide like this then container will detect BeanPostProcessor implementation automatically and IOCContainer will execute the BeanPostProcessor methods in the respective locations of the beans lifecycle.

EX: To Perform initialization ,container will perform
1.Initialization through <property> tags, that is ,setXXX() method.
2.initializing Bean through @PostConstruct
3.Initializing Bean through afterPropertiesSet() from InitializingBean.
4.Initializing Bean through custom init() method.

BeanPostProcessorImpl.java

```
package com.durgasoft.processor;
import org.springframework.beans.BeansException;
import org.springframework.beans.factory.config.BeanPostProcessor;
import com.durgasoft.beans.Customer;
public class BeanPostProcessorImpl implements BeanPostProcessor {
@Override
public Object postProcessAfterInitialization(Object bean, String name) throws
BeansException {
System.out.println("postProcessAfterInitialization()");
Customer cust=(Customer)bean;
cust.setCmobile("91-9988776655");
return cust;
}
@Override
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public Object postProcessBeforeInitialization(Object bean, String name) throws  
BeansException {  
System.out.println("postProcessBeforeInitialization()");  
Customer cust=(Customer)bean;  
cust.setCemail("durga@durgasoft.com");  
return cust;  
}  
}
```

Customer.java

```
package com.durgasoft.beans;  
public class Customer {  
    private String cid;  
    private String cname;  
    private String caddr;  
    private String cemail;  
    private String cmobile;  
  
    public Customer(){  
        System.out.println("Customer Bean Object Creating");  
    }  
    public void init(){  
        System.out.println("Customer Bean Object Initialization through init()  
method");  
    }  
    public void destroy(){  
        System.out.println("Customer Object Destroying through destroy()  
method");  
    }  
  
    setXXX()  
    getXXX()  
    public void getCustomerDetails(){  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        System.out.println("Customer Details");
        System.out.println("-----");
        System.out.println("Customer Id   :" +cid);
        System.out.println("Customer Name  :" +cname);
        System.out.println("Customer Address:" +caddr);
        System.out.println("Customer Email  :" +cemail);
        System.out.println("Customer Mobile :" +cmobile);
    }
}
```

applicationContext.xml

```
<beans>
    <bean id="cust" class="com.durgasoft.beans.Customer" init-method="init"      destroy-
method="destroy">
        <property name="cid" value="C-111"/>
        <property name="cname" value="Durga"/>
        <property name="caddr" value="Hyd"/>
    </bean>
    <bean class="com.durgasoft.processor.BeanPostProcessorImpl"/>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.support.AbstractApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Customer;
public class Test {
    public static void main(String[] args) throws Exception {
        AbstractApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        Customer cust = (Customer) context.getBean("cust");
        cust.getCustomerDetails();
        context.registerShutdownHook();
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Inversion Of Control[IOC]

In general, in enterprise application if we need any service in our application then we have to put request to the service provider, where service provider has to create the required service then service provider has to send that service to the respective application.

In enterprise applications, Inversion Of Control is a design pattern or a design principle, it will make service provider to identify the required services of the enterprise applications and it will make the service provider to create and inject the required services to the application without expecting any request from the application.

IOC is available in the following two forms.

1. Dependency Lookup
2. Dependency Injection

1. Dependency Lookup

In Dependency Lookup, Service Provider will create the services and maintained that services either in the registry softwares or in the containers , where we have to perform lookup operations inorder to get the required services.

There are two ways to achieve Dependency lookup.

1. Dependency Pull
2. Contextualized Dependency Lookup

1. Dependency Pull:

In Dependency Pull, Service Provider will create services and it will keep those services in any registry softwares, where we have to perform lookup operation in order to get the required services, that is, pull the required services from registry softwares.

EX1: When we start application Servers like Weblogic, JBOSS , Websphere and Glassfish , automatically, Application Servers will create Datasource object and Application Servers

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

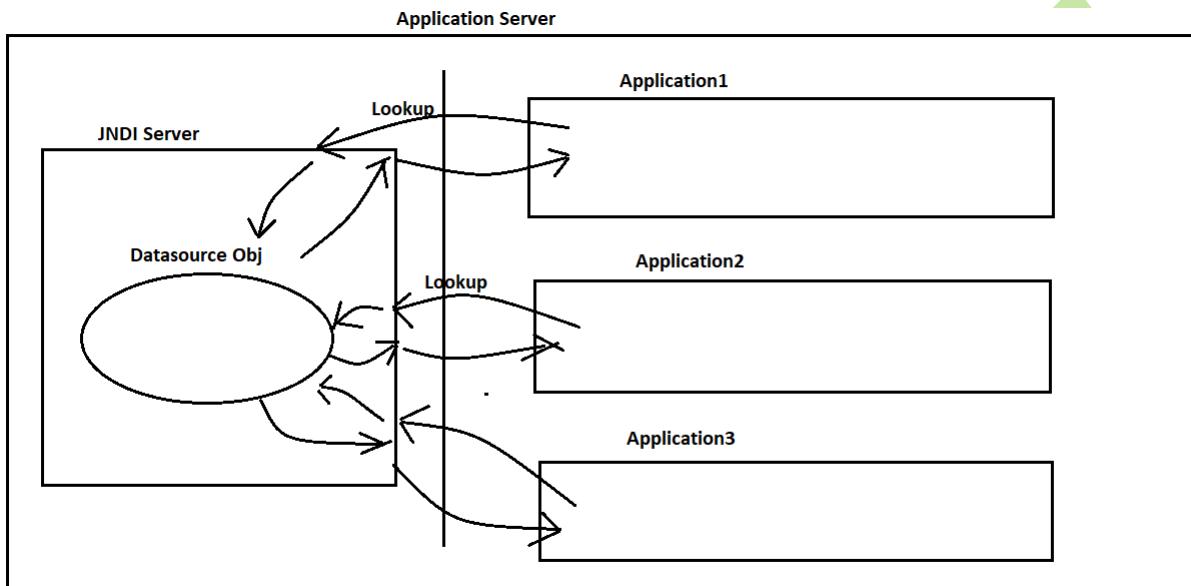
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

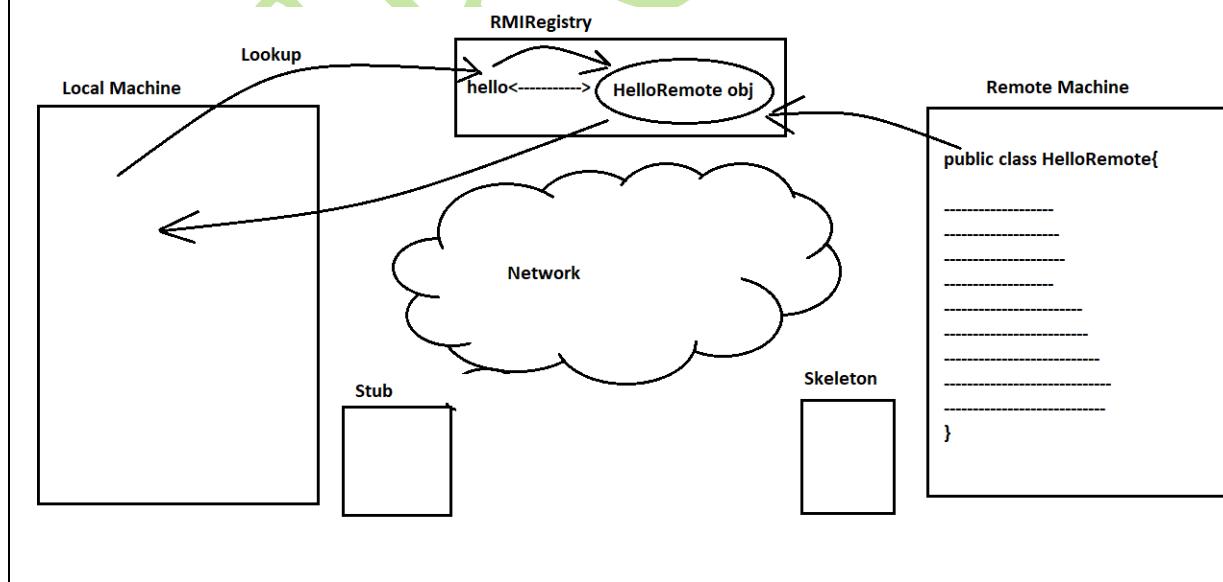


BY NAGOOR BABU

will keep that Datasource object in JNDI Server as per the server settings which we made in application Server. In this context, we have to perform lookup(--) operation over JNDI server on the basis of the logical name inorder to get the required Datasource object.



EX2: IN RMI Applications, User defined Registry Application will create Remote object and it will keep that Remote object in RMIREGISTRY with a particular logical name , here to get the required Remote object in Client Application we have to perform lookup operation over RMIREGISTRY on the basis of the logical name.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

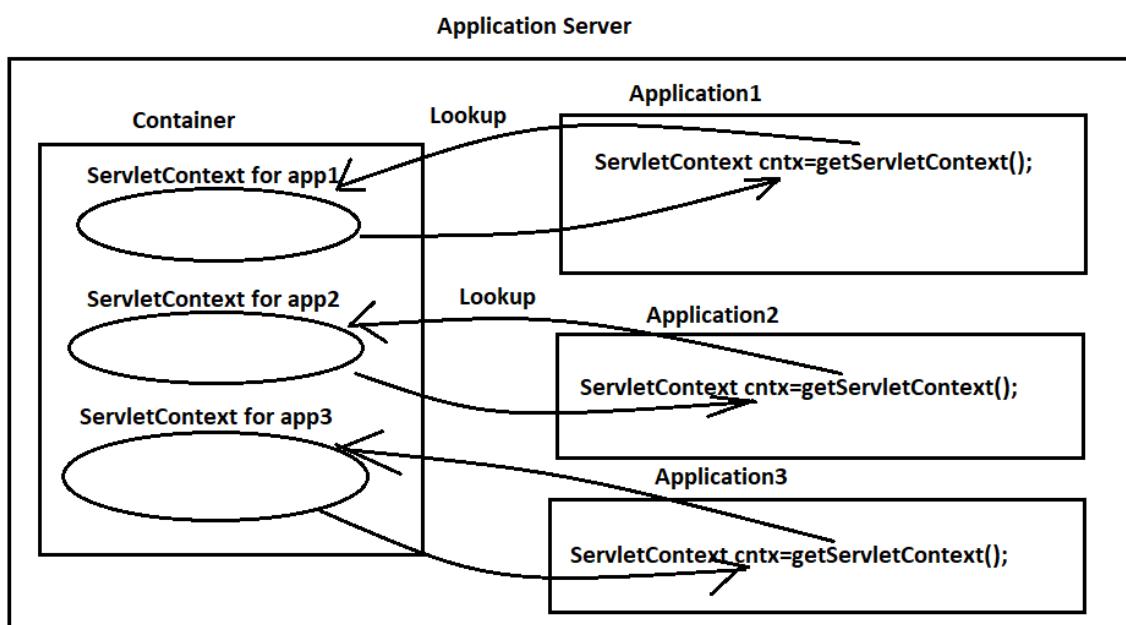


BY NAGOOR BABU

2. Contextualized Dependency Lookup

IN this mechanism, Service Provider will create the services and manage services , in this context, we have to perform lookup operation inorder to get the required services from Service Provider.

EX: In general, in web applications, when we start application Server then container will recognize all the web applications and container will deploy all the web applications into the server, when web applications are deployed in server , container will create ServletContext object automatically and Container will manage ServletContext object , in this context , to get ServletContext object we have to use `getServletContext()` method from GenericServlet directly with out using any reference variable, from ServletConfig object reference and from ServletRequest object reference, that is, performing lookup operation in Container to get ServletContext object.



2. Dependency Injection

IN this mechanism, Service Provider will create the services and inject the required services to the application directly with out performing lookup operations and with out getting request from client.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

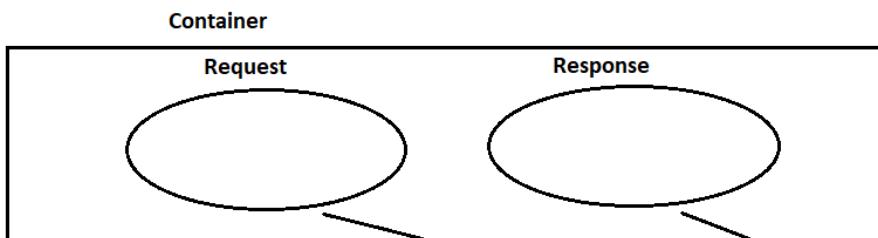
FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX: In web applications, when we submit request to a particular Servlet, Container will execute the requested Servlet by performing the Servlet life cycle actions like Servlet Loading, Servlet Instantiation, Servlet Initialization, Request Processing and Servlet Deinstantiation. In Servlet initialization phase, container will create ServletConfig object and Container will inject ServletConfig object to the Servlet program through init(--) method, in Request Processing phase, Container has to create request and response objects and Container has to inject request and response objects to the Servlet program through service() method.

```
public class MyServlet extends HttpServlet{
    ServletConfig config;
    public void init(ServletConfig config) throws ServletException{
        this.config=config;
    }
    public void service(ServletRequest request, ServletResponse response) throws ServletException, IOException{
        -----
    }
}
```



```
public class MyServlet extends HttpServlet{
    public void doXXX(HttpServletRequest req, HttpServletResponse res){
        -----
        -----
        -----
        -----
    }
}
```

There are two ways to achieve dependency injection.

1. Constructor Dependency Injection
2. Setter Method Dependency Injection

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. Constructor Dependency Injection

If we inject dependent values to the Bean object through Constructor then this type of Dependency Injection is called as "Constructor Dependency Injection".

If we want to use constructor dependency injection in Spring applications , first we have to declare the respective parameterized constructor in the corresponding bean class then we have to declare that constructor arguments in spring configuration file by using the following tags.

```
<beans>  
  
<bean ..... >  
  <constructor-arg value="--"/>  
  <constructor-arg> value </constructor-arg>  
  
</bean>  
---  
</beans>
```

EX:

```
Course.java  
-----  
package com.durgasoft.beans;  
  
public class Course {  
    private String cid;  
    private String cname;  
    private int ccost;  
  
    public Course(String cid, String cname, int ccost){  
        this.cid=cid;  
        this.cname=cname;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        this.ccost=ccost;
    }

    public void getCourseDetails(){
        System.out.println("Course Details");
        System.out.println("-----");
        System.out.println("Course Id  :" +cid);
        System.out.println("Course Name :" +cname);
        System.out.println("Course Cost :" +ccost);
    }
}
```

applicationContext.xml

```
<beans>
    <bean id="crs" class="com.durgasoft.beans.Course">
        <constructor-arg value="C-111"/>
        <constructor-arg value="JAVA"/>
        <constructor-arg value="5000"/>
    </bean>
</beans>
```

Test.java

```
package com.durgasoft.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Course;

public class Test {
    public static void main(String[] args) throws Exception {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
ApplicationContext context=new  
ClassPathXmlApplicationContext("applicationContext.xml");  
Course course=(Course)context.getBean("crs");  
course.getCourseDetails();  
}  
}
```

2. Setter Method Dependency Injection

If we inject dependent values to the Bean through setXXX() methods then it is called as "Setter Method Dependency Injection".

To inject primitive values and String value to the bean object then we have to use "value" attributes in <property> or <constructor-arg> tags in beans configuration file, but, If we want to inject User defined data types , that is, Object reference values then we have to use "ref" attribute in <property> tag or in <constructor-arg> tag.

EX:

Account.java

```
package com.durgasoft.beans;  
  
public class Account {  
    private String accNo;  
    private String accName;  
    private String accType;  
    private long balance;  
  
    setXXX()  
    getXXX()  
}
```

Employee.java

```
package com.durgasoft.beans;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class Employee {  
    private String eid;  
    private String ename;  
    private float esal;  
    private String eaddr;  
    private Account acc;  
  
    setXXX()  
    getXXX()  
  
    public void getEmployeeDetails(){  
  
        System.out.println("Employee Details");  
        System.out.println("-----");  
        System.out.println("Employee Id :"+eid);  
        System.out.println("Employee Name :" +ename);  
        System.out.println("Employee Salary :" +esal);  
        System.out.println("Employee Address:" +eaddr);  
        System.out.println("Account Details");  
        System.out.println("-----");  
        System.out.println("Account Number :" +acc.getAccNo());  
        System.out.println("Account Name :" +acc.getAccName());  
        System.out.println("Account Type :" +acc.getAccType());  
        System.out.println("Account Balance:" +acc.getBalance());  
    }  
}  
  
applicationContext.xml  
  
<beans>  
    <bean id="acc" class="com.durgasoft.beans.Account">  
        <property name="accNo" value="abc123"/>  
        <property name="accName" value="Durga"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftware.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="accType" value="Savings"/>
<property name="balance" value="25000"/>
</bean>
<bean id="emp" class="com.durgasoft.beans.Employee">
    <property name="eid" value="E-111" />
    <property name="ename" value="Durga"/>
    <property name="esal" value="50000"/>
    <property name="eaddr" value="Hyd"/>
    <property name="acc" ref="acc"/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Employee;
public class Test {
    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
        ClassPathXmlApplicationContext("applicationContext.xml");
        Employee emp = (Employee) context.getBean("emp");
        emp.getEmployeeDetails();
    }
}
```

Different Types of Elements Injection:

In Spring applications, if we want to inject User defined data types then we have to use either "ref" attribute in `<property>` and `<constructor-arg>` tags or we have to use `<ref>` nested tag under `<property>` and `<constructor-arg>` tags

EX:

```
<beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean id="--" class="--">
  <property name="--" ref="--"/>
    <ref bean="--"/>
  </property>

</bean>

</beans>
```

In spring applications, if we want to inject List of elements in beans then we have to declare the corresponding property as java.util.List and we have to provide values in configuration file by using `<list>` tag in `<property>` tag or in `<constructor-arg>` tag.

EX:

```
---
<beans>
  <bean id="---" class="--">
    <property name="--">
      <list>
        <value>value1</value>
        <value>value2</value>
        ---
        ---
      </list>
    </property>
  </bean>
</beans>
```

In Spring applications, if we want to inject Set of elements in Bean object then we have to declare the corresponding property as java.util.Set and we have to provide values in configuration file by using `<set>` tag under `<property>` tag or `<constructor-arg>` tag.

EX:

```
---
<beans>
  <bean id="---" class="--">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="--">
  <set>
    <value>value1</value>
    <value>value2</value>
    ---
    ---
  </set>
</property>
</bean>
</beans>
```

In Spring applications, if we want to inject Map of elements in Bean object then we have to declare the corresponding property as `java.util.Map` and we have to provide Key-Value pairs in configuration file by using `<map>` and `<entry>` tags under `<property>` tag or `<constructor-arg>` tag.

EX:

```
---
<beans>
  <bean id="--" class="--">
    <property name="--">
      <map>
        <entry key="key1" value="value1"/>
        <entry key="key2" value="value2"/>
        ---
      </map>
    </property>
  </bean>
</beans>
```

In Spring applications, if we want to inject Properties of elements in Bean object then we have to declare the corresponding property as `java.util.Properties` and we have to provide Key-Value pairs in configuration file by using `<props>` and `<prop>` tags under `<property>` tag or `<constructor-arg>` tag.

EX:

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
-->
<beans>
<bean id="--" class="--">
<property name="--">
<props>
<prop key="key1"> value1</prop>
<prop key="key2"> value2</prop>
<!--
</props>
</property>
</bean>
</beans>
```

Example:**Student.java**

```
package com.durgasoft.beans;

import java.util.List;
import java.util.Map;
import java.util.Properties;
import java.util.Set;

public class Student {
    private String sid;
    private String sname;
    private Address saddr;
    private List<String> squal;
    private Set<String> scourses;
    private Map<String, String> scourses_And_Faculty;
    private Properties course_And_Cost;
    setXXX()
    getXXX()
}
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public void getStudentDeails(){
System.out.println("Student Details");
System.out.println("-----");
System.out.println("Student Id    :" + sid);
System.out.println("Student Name   :" + sname);
System.out.println("Student Address :" + saddr);
System.out.println("Student Qualifications :" + squal);
System.out.println("Student Courses  :" + scourses);
System.out.println("Student Courses And Faculty :" + scourses_And_Faculty);
System.out.println("Student Courses And Cost   :" + scourse_And_Cost);
}
}
```

Address.java

```
package com.durgasoft.beans;
public class Address {
    private String pno;
    private String street;
    private String city;
    private String country;

    setXXX()
    getXXX()

    public String toString(){
        return pno + "," + street + "," + city + "," + country;
    }
}
```

applicationContext.xml

```
<beans>
    <bean id="addr" class="com.durgasoft.beans.Address">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="pno" value="202"/>
<property name="street" value="M G Road"/>
<property name="city" value="Banglore"/>
<property name="country" value="India"/>
</bean>
<bean id="std" class="com.durgasoft.beans.Student">
    <property name="sid" value="S-111"/>
    <property name="sname" value="Durga"/>
    <property name="saddr">
        <ref bean="addr"/>
    </property>
    <property name="squal">
        <list>
            <value>BTech</value>
            <value>MTech</value>
            <value>PHD</value>
        </list>
    </property>
    <property name="scourses">
        <set>
            <value>Core Java</value>
            <value>Adv Java</value>
            <value>Spring</value>
            <value>Hibernate</value>
            <value>WebServices</value>
        </set>
    </property>
    <property name="scourses_And_Faculty">
        <map>
            <entry key="Core Java" value="Ratan"/>
            <entry key="Adv Java" value="Durga"/>
            <entry key="Spring" value="Sriman"/>
            <entry key="Hibernate" value="Naveen"/>
            <entry key="Webservices" value="Naidu"/>
        </map>
    </property>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</map>
</property>
<property name="scourse_And_Cost">
    <props>
        <prop key="Core Java">1500</prop>
        <prop key="Adv Java">2000</prop>
        <prop key="Spring">3000</prop>
        <prop key="Hibernate">3000</prop>
        <prop key="Webservices">3000</prop>
    </props>
</property>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Student;
public class Test {
    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
        ClassPathXmlApplicationContext("applicationContext.xml");
        Student std = (Student) context.getBean("std");
        std.getStudentDeails();
    }
}
```

Circular Dependency Injection:

In Spring applications, if more than one bean objects are depending on each other through constructor dependency injection then it is called as Circular Dependency Injection, which is

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

not supported by Spring framework, it able to rise an exception like "
`org.springframework.beans.factory.BeanCurrentlyInCreationException`"

Ex:

Student.java

```
package com.durgasoft.beans;

public class Student {
    Branch branch;
    public Student(Branch branch) {
        this.branch=branch;
    }
    public String getStudentName(){
        return "Durga";
    }
}
```

Branch.java

```
package com.durgasoft.beans;

public class Branch {
    Student student;
    public Branch(Student student) {
        this.student=student;
    }
    public String getBranchName(){
        return "S R Nagar";
    }
}
```

applicationContext.xml

```
<beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean id="student" class="com.durgasoft.beans.Student">
    <constructor-arg ref="branch"/>
</bean>
<bean id="branch" class="com.durgasoft.beans.Branch">
    <constructor-arg ref="student"/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Branch;
import com.durgasoft.beans.Student;

public class Test {
    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
        ClassPathXmlApplicationContext("applicationContext.xml");
        Student std = (Student) context.getBean("student");
        System.out.println(std.getStudentName());
        Branch branch = (Branch) context.getBean("branch");
        System.out.println(branch.getBranchName());
    }
}
```

In Spring applications, if we want to resolve Circular Dependency Injection then we have to use Setter Method dependency Injection instead of Constructor Dependency Injection.

Student.java

```
package com.durgasoft.beans;

public class Student {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
Branch branch;
setXXX()
getXXX()
}
```

Branch.java

```
package com.durgasoft.beans;

public class Branch {
    Student student;
    setXXX()
    getXXX()
}
```

applicationContext.xml

```
<beans>
    <bean id="student" class="com.durgasoft.beans.Student">
        <property name="branch" ref="branch"/>
    </bean>
    <bean id="branch" class="com.durgasoft.beans.Branch">
        <property name="student" ref="student"/>
    </bean>
</beans>
```

Test.java

```
package com.durgasoft.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Branch;
import com.durgasoft.beans.Student;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class Test {  
    public static void main(String[] args) throws Exception {  
        ApplicationContext context = new  
        ClassPathXmlApplicationContext("applicationContext.xml");  
        Student std = (Student) context.getBean("student");  
        System.out.println(std.getStudentName());  
        Branch branch = (Branch) context.getBean("branch");  
        System.out.println(branch.getBranchName());  
    }  
}
```

Q) In Spring applications, if we provide both Setter method dependency injection and Constructor dependency injection to a single bean then what will happen in Spring Application?

Ans:

If we provide both Constructor dependency injection and Setter method dependency injection to a single bean then IOC Container will perform constructor dependency injection first at the time of creating Bean object , after that, IOC Container will perform Setter method dependency injection, that is, Constructor Dependency Injection provided values are overridden with setter method dependency injection provided values, finally, Bean object is able to manage Setter method dependency Injection provided values.

Example:

Student.java

```
package com.durgasoft.beans;  
public class Student {  
    private String sid;  
    private String sname;  
    private String saddr;  
    public Student(String sid, String sname, String saddr){
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
this.sid=sid;
this.sname=sname;
this.saddr=saddr;
System.out.println("Student---Constructor");
}

setXXX()
getXXX()

public void getStudentDetails(){
    System.out.println("Student Details");
    System.out.println("-----");
    System.out.println("Student Id   :" +sid);
    System.out.println("Student Name  :" +sname);
    System.out.println("Student Address :" +saddr);
}
}
```

applicationContext.xml

```
<beans>
    <bean id="std" class="com.durgasoft.beans.Student">
        <constructor-arg index="0" value="S-111"/>
        <constructor-arg index="1" value="AAA"/>
        <constructor-arg index="2" value="Hyd"/>

        <property name="sid" value="S-222"/>
        <property name="sname" value="BBB"/>
        <property name="saddr" value="Sec"/>
    </bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Student;
public class Test {
public static void main(String[] args) {
ApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
Student std=(Student)context.getBean("std");
std.getStudentDetails();
}
}
```

Q) What are the differences between Constructor Dependency Injection and Setter Method Dependency Injection?

Ans:

1. In Constructor dependency injection, dependent values injected through a particular constructor.

In Setter method dependency injection, dependent values are injected through properties respective setXXX() methods.

2. In Constructor Dependency Injection **readability is not good**, because, in Constructor dependency injection we are unable to identify to which property we are injecting dependent values.

In setter method Dependency injection **Readability is very good**, because, in Setter method Dependency injection we are able to identify that to property we are able to inject the dependent values.

3. In Constructor Dependency injection , dependency injection is possible when all dependent objects are getting ready, if dependent objects are not ready then Constructor dependency injection is not possible.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In Setter method dependency injection, even though dependent values are not ready, Setter method dependency injection will be performed.

4.In case of constructor dependency injection ,partial dependency injection is not possible, because, we have to access the constructor by passing the required no of parameter values.

In case of setter method dependency injection, partial dependency injection is possible , because, we are able to access setXXX() method individually.

5.IN case of constructor dependency injection, it is not simple to change the values in bean object.

In case of Setter method dependency injection , it is very simple to change the values in bean object.

6.In Constructor dependency injection, for every change on values a new bean object is created, because, for every change we have to call constructor explicitly.

In Setter method dependency injection, for every change on values new object is not created, because, for every change we can access setXXX() method explicitly.

7.Constructor dependency injection will make the bean object as "Immutable Object".

Setter method dependency injection will make the bean object as "mutable Object".

8.If we provide both Constructor and setter method dependency injection to a single bean object then setter method dependency injection overrides constructor dependency injection, but, constructor dependency injection is not overriding setter cmethod dependency injection.

9.Constructor dependency injection may provide circular dependency injection.

Setter method dependency injection will not provide circular dependency injection.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

10. Constructor dependency injection will give guarantee for dependency injection.

Setter method dependency injection will not give guarantee for dependency injection.

11. In Spring applications, if we have more no of elements to inject then it is suggestible to use Constructor dependency injection instead of setter method dependency injection.

when we have less no of properties there setter method dependency injection is better.

P-Namespace and C-Namespace :

p-Namespace:

IN general, in setter method dependency injection, to specify dependent values in spring configuration file we have to use <property> tags as per the no of properties in the bean class. In this context, to remove <property> tags and to provide dependent values as attributes in <bean> tag in spring configuration file we have to use "P-Namespace".

Note: To use p-namespace in spring configuration file we have to define "p" namespace in XSD like below.

```
xmlns:p="http://www.springframework.org/schema/p"
```

To provide value as attribute by using "p" namespace in <bean> tag we have to use the following syntax.

```
<bean id="--" class="--" p:prop_Name="value" p:prop_Name="value".../>
```

If we want to specify object reference variable as dependent value the we have to use "-ref" along with property.

```
<bean id="--" class="--" p:prop_Name-ref="ref"/>
```

C-Namespace:

In general, in constructor dependency injection, to specify dependent values in spring configuration file we have to use <constructor-arg> tags as per the no of parameters

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

which we defined in the bean class constructor . In this context, to remove <constructor-arg> tags and to provide dependent values as attributes in <bean> tag in spring configuration file we have to use "C-Namespace".

Note: To use c-namespace in spring configuration file we have to define "c" namespace in XSD like below.

```
xmlns:c="http://www.springframework.org/schema/c"
```

To provide value as attribute by using "c" namespace in <bean> tag we have to use the following syntax.

```
<bean id="--" class="--" c:arg_Name="value" c:arg_Name="value".../>
```

If we want to specify object reference variable as dependent value then we have to use "-ref" along with argument_Name.

```
<bean id="--" class="--" c:arg_Name-ref="ref"/>
```

If we want to specify dependent values in beans configuration file on the basis of index values then we have to use xml code like below.

```
<bean id="--" class="--" c:_0="val1" c:_1="val2"...c:_4-ref="ref"/>
```

Example:

Employee.java

```
package com.durgasoft.beans;

public class Employee {
    private String eid;
    private String ename;
    private float esal;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private Address eaddr;
```

```
setXXX()  
getXXX()
```

```
public void getEmpDetails(){  
    System.out.println("Employee Details");  
    System.out.println("-----");  
    System.out.println("Employee Id :"+eid);  
    System.out.println("Employee Name :"+ename);  
    System.out.println("Employee Salary :"+esal);  
    System.out.println();  
    System.out.println("Employee Address Details");  
    System.out.println("-----");  
    System.out.println("House Number :"+eaddr.getHno());  
    System.out.println("Street :"+eaddr.getStreet());  
    System.out.println("City :"+eaddr.getCity());  
    System.out.println("State :"+eaddr.getState());  
}
```

Address.java

```
package com.durgasoft.beans;  
  
public class Address {  
    String hno;  
    String street;  
    String city;  
    String state;  
  
    setXXX()  
    getXXX()  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Student.java

```
package com.durgasoft.beans;

public class Student {
    String sid;
    String sname;
    String saddr;
    Course crs;
    public Student(String sid, String sname, String saddr, Course
        crs) {
        this.sid=sid;
        this.sname=sname;
        this.saddr=saddr;
        this.crs=crs;
    }
    public void getStudentDetails(){
        System.out.println("Student Details");
        System.out.println("-----");
        System.out.println("Student Id :"+sid);
        System.out.println("Student Name :"+sname);
        System.out.println("Student Address :"+saddr);
        System.out.println();
        crs.getCourseDetails();
    }
}
```

Course.java

```
package com.durgasoft.beans;

public class Course {
    String cid;
    String cname;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
int ccost;

public Course(String cid, String cname, int ccost) {
    this.cid=cid;
    this.cname=cname;
    this.ccost=ccost;
}
public void getCourseDetails(){
    System.out.println("Course Details");
    System.out.println("-----");
    System.out.println("Course Id :"+cid);
    System.out.println("Course Name :"+cname);
    System.out.println("Course Cost :"+ccost);
}
}
```

applicationContext.xml

```
<beans -----
    xmlns:p="http://www.springframework.org/schema/p"
    xmlns:c="http://www.springframework.org/schema/c"
    ----
    >
    <bean id="emp" class="com.durgasoft.beans.Employee"
        p:eid="E-111" p:ename="AAA" p:esal="15000" p:addr-ref="addr"/>
    <bean name="addr" class="com.durgasoft.beans.Address"
        p:hno="23/3rt" p:street="M G Road" p:city="Hyd" p:state="Tel"/>

    <bean id="std" class="com.durgasoft.beans.Student" c:sid="S-111"
        c:sname="AAA" c:saddr="Hyd" c:crs-ref="crs"/>
    <bean id="crs" class="com.durgasoft.beans.Course" c:_0="C-111"
        c:_1="JAVA" c:_2="10000"/>
</beans>
```

Test.java

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
package com.durgasoft.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Employee;
import com.durgasoft.beans.Student;

public class Test {
public static void main(String[] args) {
ApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
Employee emp=(Employee)context.getBean("emp");
emp.getEmpDetails();
System.out.println();
Student std=(Student)context.getBean("std");
std.getStudentDetails();
}
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Beans Autowiring/Beans Collaboration

In general, in spring applications, if we want to inject dependent values in setter method dependency injection or in constructor dependency injection we have to use <property> or <constructor-arg> tags under <bean> tag. If we want to inject simple values like primitive values, string values then we have to use "value" attribute and if we want to inject Secondary data type elements like Objects then we have to use "ref" attribute or we have to use <ref> tag in beans configuration file.

In spring applicatins , if we want to inject dependent bean objects to another bean object automatically with out providing <property> tags and <constructor-arg> tags then we have to use "Autowiring" feature.

"Autowiring" feature of spring framework will make the IOC Container to inject dependent objects to the bean objects automatically on the basis of the properties names or on the basis of properties types with out checking <property> tags and <constructor-arg> tags.

There are four ways to manage autowiring in Spring applications.

- 1.XML Based Autowiring
- 2.Annotation Based Autowiring
- 3.Auto-Discovery[Stereotype Types]
- 4.Java Based Autowiring

1. XML Based Auto wiring

In this approach, If we want to provide autowiring in spring applications then we have to use "autowire" attribute in <bean> tag

EX:

```
<bean id="--" class="--" autowire="value">
```

Here value may be either of the following "autowiring modes".

- 1.no
- 2.byName

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3.byId
4.constructor

1.no

It is representing "no" autowiring for the beans injection, we must provide explicit configuration for the beans injection.

it is not representing any of the autowiring mechanisms, it specifies go for explicit wiring.

2.byName

It will provide autowiring on the basis of the properties names. In this autowiring mode, IOC Container will search for dependent bean objects by matching bean properties names with the identity values of the beans configuration in spring configuration file.

3.byId

It will provide autowiring on the basis of the properties data types. In this autowiring mode, IOC Container will identify the dependent bean objects by matching properties data types with the bean data types[class attribute values] in bean configuration.

Note: In Beans configuration file, only one bean definition must be existed with the same type , if we provide more than one bean configuration with the same type in beans configuration file then IOC Container will rise an exception.

if you dont want to inject other bean object then used ""autowire-candidate="false" "" in case of type imbugity

4.constructor problem in ByType.

It is same as "byType" autowiring mode, but, "byType" autowiring will provide setter method dependency injection and "constructor" autowiring mode will provide constructor dependency injection on the basis of the types.

Example:

Address.java

```
package com.durgasoftware.beans;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftlinetraining@gmail.com

WEBSITE: www.durgasoftwareonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class Address {  
    private String hno;  
    private String street;  
    private String city;  
    private String state;  
  
    setXXX()  
    getXXX()  
}
```

Account.java

```
package com.durgasoft.beans;  
public class Account {  
    private String accNo;  
    private String accName;  
    private String accType;  
    private long balance;  
  
    setXXX()  
    getXXX()  
}
```

Employee.java

```
package com.durgasoft.beans;  
public class Employee {  
    private String eid;  
    private String ename;  
    private Address eaddr;  
    private Account eacc;  
  
    setXXX()
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

getXXX()

```
public void getEmpDetails(){
    System.out.println("Employee Details");
    System.out.println("-----");
    System.out.println("Employee Id : "+eid);
    System.out.println("Employee Name : "+ename);
    System.out.println();
    System.out.println("Employee Address Details");
    System.out.println("-----");
    System.out.println("House Number:"+eaddr.getHno());
    System.out.println("Street : "+eaddr.getStreet());
    System.out.println("City : "+eaddr.getCity());
    System.out.println("State : "+eaddr.getState());
    System.out.println();
    System.out.println("Employee Account Details");
    System.out.println("-----");
    System.out.println("Account Number : "+eacc.getAccNo());
    System.out.println("Account Name : "+eacc.getAccName());
    System.out.println("Account Type : "+eacc.getAccType());
    System.out.println("Account Balance: "+eacc.getBalance());
}
```

applicationContext.xml

```
<beans>
    <bean id="eaddr" class="com.durgasoft.beans.Address">
        <property name="hno" value="23/3rt"/>
        <property name="street" value="PS Road"/>
        <property name="city" value="Hyd"/>
        <property name="state" value="Tel"/>
    </bean>
    <bean id="eacc" class="com.durgasoft.beans.Account">
        <property name="accNo" value="abc123"/>
        <property name="accName" value="Durga"/>
    </bean>
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="accType" value="Savings"/>
<property name="balance" value="20000"/>
</bean>
<bean id="emp" class="com.durgasoft.beans.Employee" autowire="byName">
    <property name="eid" value="E-111"/>
    <property name="ename" value="Durga"/>
    <!--
        <property name="eaddr" ref="eaddr"/>
        <property name="eacc" ref="eacc"/>
    -->
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Employee;
public class Test {
    public static void main(String[] args){
        ApplicationContext context=new
        ClassPathXmlApplicationContext("applicationContext.xml");
        Employee emp=(Employee)context.getBean("emp");
        emp.getEmpDetails();
    }
}
```

If we want to provide example for "constructor" autowiring then we have to use the following components in the above example

Example:

Address.java

```
package com.durgasoft.beans;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class Address {  
    private String hno;  
    private String street;  
    private String city;  
    private String state;  
  
    setXXX()  
    getXXX()  
}
```

Account.java

```
package com.durgasoft.beans;  
public class Account {  
    private String accNo;  
    private String accName;  
    private String accType;  
    private long balance;  
  
    setXXX()  
    getXXX()  
}
```

Employee.java

```
public class Employee {  
    private String eid;  
    private String ename;  
    private Address eaddr;  
    private Account eacc;  
  
    public Employee(String eid, String ename, Address eaddr, Account eacc ){  
        this.eid=eid;  
        this.ename=ename;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

        this.eaddr=eaddr;
        this.eacc=eacc;
    }
    public void getEmpDetails(){
        System.out.println("Employee Details");
        System.out.println("-----");
        System.out.println("Employee Id   :" + eid);
        System.out.println("Employee Name  :" + ename);
        System.out.println();
        System.out.println("Employee Address Details");
        System.out.println("-----");
        System.out.println("House Number:" + eaddr.getHno());
        System.out.println("Street           :" + eaddr.getStreet());
        System.out.println("City            :" + eaddr.getCity());
        System.out.println("State           :" + eaddr.getState());
        System.out.println();
        System.out.println("Employee Account Details");
        System.out.println("-----");
        System.out.println("Account Number :" + eacc.getAccNo());
        System.out.println("Account Name   :" + eacc.getAccName());
        System.out.println("Account Type   :" + eacc.getAccType());
        System.out.println("Account Balance:" + eacc.getBalance());
    }
}

```

applicationContext.xml

```

<beans>
    <bean id="eaddr" class="com.durgasoft.beans.Address">
        same as above
    </bean>
    <bean id="eacc" class="com.durgasoft.beans.Account">
        same as above
    </bean>
    <bean id="emp" class="com.durgasoft.beans.Employee" autowire="constructor">

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<constructor-arg name="eid" value="E-111"/>
<constructor-arg name="ename" value="Durga"/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Employee;
public class Test {
public static void main(String[] args){
ApplicationContext context=new
ClassPathXmlApplicationContext("applicationContext.xml");
Employee emp=(Employee)context.getBean("emp");
emp.getEmpDetails();
}
}
```

Note: If we want to block any bean object in autowiring then we have to use "autowire-candidate" attribute with "false" value in <bean> tag in beans configuration file.

EX:

```
<beans>
<bean id="scourse" class="com.durgasoft.beans.Course"
autowire-candidate="false">
---
</bean>
<bean id="student" class="com.durgasoft.beans.Student" autowire="byType">
---
</bean>
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Annotations for Autowiring:

To implement Autowiring in Spring applications without providing autowiring configuration in spring configuration file , we have to use the following annotations provided by spring framework.

- 1.@Required
- 2.@Autowired
- 3.@Qualifier

1.@Required

This annotation will make IOC Container to inject a particular bean object in another bean object is mandatory. We have to use this annotation at method level, that is, just before setXXX() method. After providing this annotation, if we are not providing the respective bean injection then IOC Container will rise an exception .

2.@Autowired

This annotation is able to represent autowiring in bean classes, it will be used at method level, field level and local variables level in constructor dependency injection.

Note: If we provide "required" argument with "false" value in @Autowired annotation then it is not required to use @Required annotationi.

Note: This annotation is following "byType" autowiring internally in spring applications, If we want to use this annotation then we must have only one bean configuration with the respective type in configuration file, if we have more than one bean configuration with the same type then IOC Container will rise an exception.

3.@Qualifier

In the case of "byType" autowiring mode, that is, in the case of @Autowired annotation configuration file must provide only one bean configuration with the respective type, if we provide more than one bean configuration with the same type then IOC Container will rise an exception. In this context, to resolve the ambiguity of beans injection we

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

have to use "@Qualifier" annotation, it will be used to specify a particular bean object among the multiple beans of the same type for injection.

EX: @Qualifier("bean_Identity")

Example:

Student.java

```
package com.durgasoft.beans;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.beans.factory.annotation.Required;

public class Student {
    private String sid;
    private String sname;
    private Course scourse;

    public String getSid() {
        return sid;
    }
    public void setSid(String sid) {
        this.sid = sid;
    }
    public String getSname() {
        return sname;
    }
    public void setSname(String sname) {
        this.sname = sname;
    }

    public Course getScourse() {
        return scourse;
    }
    @Autowired(required=true)
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

//@Required
@Qualifier("adv_Java")
public void setScourse(Course scourse) {
    this.scourse = scourse;
}
public void getStudentDetails(){
    System.out.println("Student Details");
    System.out.println("-----");
    System.out.println("Student Id :"+sid);
    System.out.println("Student Name :" +sname);
    System.out.println("Course Details");
    System.out.println("-----");
    System.out.println("Course Id :" +scourse.getCid());
    System.out.println("Course Name :" +scourse.getCname());
    System.out.println("Course Cost :" +scourse.getCCost());
}

```

Course.java

```

package com.durgasoft.beans;
public class Course {
    private String cid;
    private String cname;
    private int ccost;
    setXXX()
    getXXX()
}

```

applicationContext.xml

```

<beans ----- >
    <context:annotation-config/>
    <bean id="core_Java" class="com.durgasoft.beans.Course">
        <property name="cid" value="C-111"/>
        <property name="cname" value="Core Java"/>
        <property name="ccost" value="10000"/>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</bean>
<bean id="adv_Java" class="com.durgasoft.beans.Course">
    <property name="cid" value="C-111"/>
    <property name="cname" value="Adv Java"/>
    <property name="ccost" value="20000"/>
</bean>
<bean id="std" class="com.durgasoft.beans.Student" >
    <property name="sid" value="S-111"/>
    <property name="sname" value="Durga"/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Student;
public class Test {
    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
        ClassPathXmlApplicationContext("applicationContext.xml");
        Student std = (Student) context.getBean("std");
        std.getStudentDetails();
    }
}
```

If we want to use @Autowired annotation for constructor dependency injection then we have to use @Autowired annotation just above of the respective constructor and we have to use @Qualifier annotation along with with the Bean parameter in constructor.

Example:

Student.java

```
package com.durgasoft.beans;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.beans.factory.annotation.Required;

public class Student {
    private String sid;
    private String sname;
    private Course scourse;

    @Autowired
    public Student(String sid, String sname,
    @Qualifier("adv_Java")Course scourse){
        this.sid=sid;
        this.sname=sname;
        this.scourse=scourse;
    }
    public void getStudentDetails(){
        System.out.println("Student Details");
        System.out.println("-----");
        System.out.println("Student Id :"+sid);
        System.out.println("Student Name :" +sname);
        System.out.println("Course Details");
        System.out.println("-----");
        System.out.println("Course Id : "+scourse.getCid());
        System.out.println("Course Name :" +scourse.getCname());
        System.out.println("Course Cost :" +scourse.getCost());
    }
}
```

Course.java

```
package com.durgasoft.beans;
public class Course {
    private String cid;
    private String cname;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private int ccost;  
  
    setXXX()  
    getXXX()  
}
```

applicationContext.xml

```
<beans>  
    <context:annotation-config/>  
    <bean id="core_Java" class="com.durgasoft.beans.Course">  
        <property name="cid" value="C-111"/>  
        <property name="cname" value="Core Java"/>  
        <property name="ccost" value="10000"/>  
    </bean>  
    <bean id="adv_Java" class="com.durgasoft.beans.Course">  
        <property name="cid" value="C-111"/>  
        <property name="cname" value="Adv Java"/>  
        <property name="ccost" value="20000"/>  
    </bean>  
    <bean id="std" class="com.durgasoft.beans.Student" >  
        <constructor-arg name="sid" value="S-111"/>  
        <constructor-arg name="sname" value="Durga"/>  
    </bean>  
</beans>
```

Test.java

```
package com.durgasoft.test;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import com.durgasoft.beans.Student;  
public class Test {  
    public static void main(String[] args) throws Exception {  
        ApplicationContext context=new  
        ClassPathXmlApplicationContext("applicationContext.xml");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
Student std=(Student)context.getBean("std");
std.getStudentDetails();
}
}
```

3.Auto-Discovery[Stereo Types]

This mechanism will provide the autowiring beans objects with out using <bean> configuration in configuration file.

To use this mechanism in Spring applications then we have to use the following annotations provided by spring framework in the package "org.springframework.stereotype"

- 1.@Component: It will represent a component which is recognized by Spring Container.
- 2.@Repository: It will represent a class as Model Driven , that is, DAO.
- 3.@Service : It will represent a class as Service class.
- 4.@Controller: It will represent a class as Controller class, it will be used in Spring WEB-MVC Module.

Note: If we want to use these annotations in Spring applications then we must provide the following tag in spring configuration file.

```
<context:component-scan base-package="---"/>
```

EX:

```
---
```

```
<context:component-scan base-package="com.durgasoft.service"/>
<context:component-scan base-package="com.durgasoft.dao"/>
<context:component-scan base-package="com.durgasoft.controller"/>
```

If we provide the above tag in spring configuration file then IOC Container will scan the specified packages and recognize the classes which are annotated with @Component, @Repository, @Service and @Controller then Container will create bean objects with out checking beans configurations in configuration file.

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:

AccountDao.java

```
-----  
package com.durgasoft.dao;  
  
import com.durgasoft.dto.Account;  
  
public interface AccountDao {  
    public String create(String accNo, String accName, String accType, int balance);  
    public String search(String accNo);  
    public Account getAccount(String accNo);  
    public String update(String accNo, String accName, String accType, int balance);  
    public String delete(String accNo);  
}
```

AccountDaoImpl.java

```
-----  
package com.durgasoft.dao;  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Component;  
import org.springframework.stereotype.Repository;  
  
import com.durgasoft.dto.Account;  
  
import oracle.jdbc.pool.OracleDataSource;  
  
//@Repository("accDao")  
//@Component("accDao")  
public class AccountDaoImpl implements AccountDao {  
    String status = "";
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
@Autowired(required=true)
private OracleDataSource dataSource;

@Override
public String create(String accNo, String accName, String accType, int balance) {
    try {

        Connection con = dataSource.getConnection();
        PreparedStatement pst = con.prepareStatement("select * from
account where accNo=?");
        pst.setString(1, accNo);
        ResultSet rs = pst.executeQuery();
        boolean b = rs.next();
        if(b == true) {
            status="existed";
        }else {
            pst = con.prepareStatement("insert into account
values(?,?,?,?,?)");
            pst.setString(1, accNo);
            pst.setString(2, accName);
            pst.setString(3, accType);
            pst.setInt(4, balance);
            pst.executeUpdate();
            status="success";
        }
    } catch (Exception e) {
        status = "failure";
        e.printStackTrace();
    }
    return status;
}

@Override
public String search(String accNo) {
    try {
        Connection con = dataSource.getConnection();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
PreparedStatement pst = con.prepareStatement("select * from account where accNo = ?");  
pst.setString(1, accNo);  
ResultSet rs = pst.executeQuery();  
boolean b = rs.next();  
if(b == true) {  
    status =  
    "[ACCNO:"+rs.getString("ACCNO")+","ACCNAME:"+rs.getString("ACCNAME")+",ACCTYP  
E:"+rs.getString("ACCTYPE")+",BALANCE:"+rs.getInt("BALANCE")+"]";  
} else {  
    status = "Account Not Existed";  
}  
} catch (Exception e) {  
    e.printStackTrace();  
}  
return status;  
}  
  
@Override  
public Account getAccount(String accNo) {  
    Account acc = null;  
    try {  
        Connection con = dataSource.getConnection();  
        PreparedStatement pst = con.prepareStatement("select * from account where accNO = ?");  
        pst.setString(1, accNo);  
        ResultSet rs = pst.executeQuery();  
        boolean b = rs.next();  
        if(b == true) {  
            acc = new Account();  
            acc.setAccNo(rs.getString("ACCNO"));  
            acc.setAccName(rs.getString("ACCNAME"));  
            acc.setAccType(rs.getString("ACCTYPE"));  
            acc.setBalance(rs.getInt("BALANCE"));  
        } else {  
            acc = null;  
        }  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
    return acc;  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return acc;
}
@Override
public String update(String accNo, String accName, String accType, int balance) {
    try {
        Connection con = dataSource.getConnection();
        PreparedStatement pst = con.prepareStatement("update account set
ACCNAME = ?, ACCTYPE = ?, BALANCE = ? where ACCNO = ?");
        pst.setString(1, accName);
        pst.setString(2, accType);
        pst.setInt(3, balance);
        pst.setString(4, accNo);
        pst.executeUpdate();
        status = "success";
    } catch (Exception e) {
        status = "failure";
        e.printStackTrace();
    }
    return status;
}
@Override
public String delete(String accNo) {
    try {
        Connection con = dataSource.getConnection();
        PreparedStatement pst = con.prepareStatement("select * from
account where accNO = ?");
        pst.setString(1, accNo);
        ResultSet rs = pst.executeQuery();
        boolean b = rs.next();
        if(b == true) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        pst = con.prepareStatement("delete from account where accNo  
= ?");  
        pst.setString(1, accNo);  
        pst.executeUpdate();  
        status = "success";  
    }else {  
        status = "notexisted";  
    }  
} catch (Exception e) {  
    status = "failure";  
    e.printStackTrace();  
}  
return status;  
}  
}
```

AccountService.java

```
-----  
package com.durgasoft.service;  
  
import com.durgasoft.dto.Account;  
  
public interface AccountService {  
    public String createAccount(String accNo, String accName, String accType, int  
balance);  
    public String searchAccount(String accNo);  
    public Account getAccount(String accNo);  
    public String updateAccount(String accNo, String accName, String accType, int  
balance);  
    public String deleteAccount(String accNo);  
}
```

AccountServiceImpl.java

```
-----  
package com.durgasoft.service;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.durgasoft.dao.AccountDao;
import com.durgasoft.dto.Account;

@Service("accService")
public class AccountServiceImpl implements AccountService {

    @Autowired(required=true)
    private AccountDao dao;
    @Override
    public String createAccount(String accNo, String accName, String accType, int
balance) {

        return dao.create(accNo, accName, accType, balance);
    }

    @Override
    public String searchAccount(String accNo) {

        return dao.search(accNo);
    }

    @Override
    public Account getAccount(String accNo) {

        return dao.getAccount(accNo);
    }

    @Override
    public String updateAccount(String accNo, String accName, String accType, int
balance) {

        return dao.update(accNo, accName, accType, balance);
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
@Override  
public String deleteAccount(String accNo) {  
    return dao.delete(accNo);  
}  
}
```

Account.java

```
-----  
package com.durgasoft.dto;  
  
public class Account {  
    private String accNo;  
    private String accName;  
    private String accType;  
    private int balance;  
  
    public String getAccNo() {  
        return accNo;  
    }  
    public void setAccNo(String accNo) {  
        this.accNo = accNo;  
    }  
    public String getAccName() {  
        return accName;  
    }  
    public void setAccName(String accName) {  
        this.accName = accName;  
    }  
    public String getAccType() {  
        return accType;  
    }  
    public void setAccType(String accType) {  
        this.accType = accType;  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
}

public int getBalance() {
    return balance;
}

public void setBalance(int balance) {
    this.balance = balance;
}

}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
                           http://www.springframework.org/schema/beans/spring-beans.xsd
                           http://www.springframework.org/schema/context
                           http://www.springframework.org/schema/context/spring-context.xsd">

    <context:component-scan base-package="com.durgasoft.service"/>
    <context:component-scan base-package="com.durgasoft.dao"/>
    <bean id="dataSource" class="oracle.jdbc.pool.OracleDataSource">
        <property name="URL" value="jdbc:oracle:thin:@localhost:1521:xe"/>
        <property name="user" value="system"/>
        <property name="password" value="durga"/>
    </bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import java.io.BufferedReader;
import java.io.InputStreamReader;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.dao.AccountDao;
import com.durgasoft.dto.Account;
import com.durgasoft.service.AccountService;

public class Test {

    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        AccountService accService =
(AccountService)context.getBean("accService");
        BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

        while(true) {
            System.out.println();
            System.out.println("Account Operations Menu");
            System.out.println("1.Create Account");
            System.out.println("2.Search Account");
            System.out.println("3.Update Account");
            System.out.println("4.Delete Account");
            System.out.println("5.Exit");
            System.out.print("Your Option :");
            int option = Integer.parseInt(br.readLine());
            String status = "";
            String accNo = "", accName = "", accType = "";
            int balance = 0;
            switch(option) {
            case 1:
                System.out.print("Account Number :");

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
accNo = br.readLine();
System.out.print("Account Name   :");
accName = br.readLine();
System.out.print("Account Type   :");
accType = br.readLine();
System.out.print("Balance      :");
balance = Integer.parseInt(br.readLine());

status = accService.createAccount(accNo, accName, accType,
balance);
if(status.equals("success")) {
    System.out.println("Account Created Successfully");
}
if(status.equals("failure")){
    System.out.println("Account Creation Failure");
}
if(status.equals("existed")) {
    System.out.println("Account Existed Already");
}
break;

case 2:
System.out.print("Account Number :");
accNo = br.readLine();
status = accService.searchAccount(accNo);
System.out.println("Account Details :" +status);
break;

case 3:
System.out.print("Account Number :");
accNo = br.readLine();
Account acc = accService.getAccount(accNo);
if(acc == null) {
    System.out.println("Status :Account Not Existed");
}else {
    Account acc_New = new Account();
    acc_New.setAccNo(accNo);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
System.out.print("Account Name : Old Value  
:"+acc.getAccName()+" New Value :");  
String accName_New = br.readLine();  
if(accName_New == null || accName_New.equals("")) {  
    acc_New.setAccName(acc.getAccName());  
}else {  
    acc_New.setAccName(accName_New);  
}  
  
System.out.print("Account Type : Old Value  
:"+acc.getAccType()+" New Value :");  
String accType_New = br.readLine();  
if(accType_New == null || accType_New.equals("")) {  
    acc_New.setAccType(acc.getAccType());  
}else {  
    acc_New.setAccType(accType_New);  
}  
  
System.out.print("Account Balance : Old Value  
:"+acc.getBalance()+" New Value :");  
String bal = br.readLine();  
if(bal == null || bal.equals("")) {  
    acc_New.setBalance(acc.getBalance());  
}else {  
    int balance_New = Integer.parseInt(bal);  
    acc_New.setBalance(balance_New);  
}  
status = accService.updateAccount(acc_New.getAccNo(),  
        acc_New.getAccName(), acc_New.getAccType(), acc_New.getBalance());  
if(status.equals("success")) {  
    System.out.println("Account Updated  
Successfully");  
}  
if(status.equals("failure")) {  
    System.out.println("Account Updation Failure");  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



```
        }
        break;
    case 4:
        System.out.print("Account Number :");
        accNo = br.readLine();
        status = accService.deleteAccount(accNo);
        if(status.equals("success")) {
            System.out.println("Account Deleted Successfully");
        }
        if(status.equals("failure")) {
            System.out.println("Account Deletion Failure");
        }
        if(status.equals("notexisted")) {
            System.out.println("Account Not Existed");
        }
        break;
    case 5:
        System.out.println("***** ThankQ for Using Account
Operations App*****");
        System.exit(0);
        break;
    default:
        System.out.println("Enter Number from 1,2,3,4 and 5");
        break;
}
```

4. Java Based Autowiring

we have to user AnnotationConfigApplicationContext(AppConfig.class)

AccountDao.java

```
| package com.durgasoft.dao;
```

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com



US NUM: 4433326786



FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import com.durgasoft.dto.Account;

public interface AccountDao {
    public String create(String accNo, String accName, String accType, int balance);
    public String search(String accNo);
    public Account getAccount(String accNo);
    public String update(String accNo, String accName, String accType, int balance);
    public String delete(String accNo);
}
```

[AccountDaoImpl.java](#)

```
package com.durgasoft.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;
import org.springframework.stereotype.Repository;

import com.durgasoft.dto.Account;

import oracle.jdbc.pool.OracleDataSource;

//@Repository("accDao")
@Component("accDao")
public class AccountDaoImpl implements AccountDao {
    String status = "";
    @Autowired(required=true)
    private OracleDataSource dataSource;

    @Override
    public String create(String accNo, String accName, String accType, int balance) {
        try {
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
Connection con = dataSource.getConnection();
PreparedStatement pst = con.prepareStatement("select * from
account where accNo=?");
pst.setString(1, accNo);
ResultSet rs = pst.executeQuery();
boolean b = rs.next();
if(b == true) {
    status="existed";
}else {
    pst = con.prepareStatement("insert into account
values(?,?,?,?,?)");
    pst.setString(1, accNo);
    pst.setString(2, accName);
    pst.setString(3, accType);
    pst.setInt(4, balance);
    pst.executeUpdate();
    status="success";
}
} catch (Exception e) {
    status = "failure";
    e.printStackTrace();
}
return status;
}

@Override
public String search(String accNo) {
try {
    Connection con = dataSource.getConnection();
    PreparedStatement pst = con.prepareStatement("select * from
account where accNo = ?");
    pst.setString(1, accNo);
    ResultSet rs = pst.executeQuery();
    boolean b = rs.next();
    if(b == true) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
status =  
"[ACCNO:"+rs.getString("ACCNO")+","ACCNAME:"+rs.getString("ACCNAME")+",ACCTYP  
E:"+rs.getString("ACCTYPE")+",BALANCE:"+rs.getInt("BALANCE")+"]";  
    }else {  
        status = "Account Not Existed";  
    }  
} catch (Exception e) {  
    e.printStackTrace();  
}  
return status;  
}  
  
@Override  
public Account getAccount(String accNo) {  
    Account acc = null;  
    try {  
        Connection con = dataSource.getConnection();  
        PreparedStatement pst = con.prepareStatement("select * from  
account where accNO = ?");  
        pst.setString(1, accNo);  
        ResultSet rs = pst.executeQuery();  
        boolean b = rs.next();  
        if(b == true) {  
            acc = new Account();  
            acc.setAccNo(rs.getString("ACCNO"));  
            acc.setAccName(rs.getString("ACCNAME"));  
            acc.setAccType(rs.getString("ACCTYPE"));  
            acc.setBalance(rs.getInt("BALANCE"));  
        }else {  
            acc = null;  
        }  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
    return acc;  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
@Override  
public String update(String accNo, String accName, String accType, int balance) {  
    try {  
        Connection con = dataSource.getConnection();  
        PreparedStatement pst = con.prepareStatement("update account set  
ACCNAME = ?, ACCTYPE = ?, BALANCE = ? where ACCNO = ?");  
        pst.setString(1, accName);  
        pst.setString(2, accType);  
        pst.setInt(3, balance);  
        pst.setString(4, accNo);  
        pst.executeUpdate();  
        status = "success";  
    } catch (Exception e) {  
        status = "failure";  
        e.printStackTrace();  
    }  
    return status;  
}  
  
@Override  
public String delete(String accNo) {  
    try {  
        Connection con = dataSource.getConnection();  
        PreparedStatement pst = con.prepareStatement("select * from  
account where accNO = ?");  
        pst.setString(1, accNo);  
        ResultSet rs = pst.executeQuery();  
        boolean b = rs.next();  
        if(b == true) {  
            pst = con.prepareStatement("delete from account where accNo  
= ?");  
            pst.setString(1, accNo);  
            pst.executeUpdate();  
            status = "success";  
        }else {  
            status = "notexisted";  
        }  
    } catch (Exception e) {  
        status = "failure";  
        e.printStackTrace();  
    }  
    return status;  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        }
    } catch (Exception e) {
        status = "failure";
        e.printStackTrace();
    }
    return status;
}

}
```

AccountService.java

```
package com.durgasoft.service;

import com.durgasoft.dto.Account;

public interface AccountService {
    public String createAccount(String accNo, String accName, String accType, int
balance);
    public String searchAccount(String accNo);
    public Account getAccount(String accNo);
    public String updateAccount(String accNo, String accName, String accType, int
balance);
    public String deleteAccount(String accNo);
}
```

AccountServiceImpl.java

```
package com.durgasoft.service;

import com.durgasoft.dto.Account;

public interface AccountService {
    public String createAccount(String accNo, String accName, String accType, int
balance);
    public String searchAccount(String accNo);
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public Account getAccount(String accNo);
    public String updateAccount(String accNo, String accName, String accType, int
balance);
    public String deleteAccount(String accNo);
}
```

AccountServiceImpl.java

```
package com.durgasoft.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.durgasoft.dao.AccountDao;
import com.durgasoft.dto.Account;

@Service("accService")
public class AccountServiceImpl implements AccountService {

    @Autowired(required=true)
    private AccountDao dao;
    @Override
    public String createAccount(String accNo, String accName, String accType, int
balance) {
        return dao.create(accNo, accName, accType, balance);
    }
    @Override
    public String searchAccount(String accNo) {
        return dao.search(accNo);
    }

    @Override
    public Account getAccount(String accNo) {
```

CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        return dao.getAccount(accNo);
    }
    @Override
    public String updateAccount(String accNo, String accName, String accType, int
balance) {
        return dao.update(accNo, accName, accType, balance);
    }

    @Override
    public String deleteAcount(String accNo) {
        return dao.delete(accNo);
    }
}
```

Account.java

```
package com.durgasoft.dto;

public class Account {
    private String accNo;
    private String accName;
    private String accType;
    private int balance;

    public String getAccNo() {
        return accNo;
    }
    public void setAccNo(String accNo) {
        this.accNo = accNo;
    }
    public String getAccName() {
        return accName;
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
    }
    public void setAccName(String accName) {
        this.accName = accName;
    }
    public String getAccType() {
        return accType;
    }
    public void setAccType(String accType) {
        this.accType = accType;
    }
    public int getBalance() {
        return balance;
    }
    public void setBalance(int balance) {
        this.balance = balance;
    }
}
```

AccountConfig.java

```
package com.durgasoft.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

import com.durgasoft.dao.AccountDao;
import com.durgasoft.dao.AccountDaoImpl;
import com.durgasoft.service.AccountService;
import com.durgasoft.service.AccountServiceImpl;

import oracle.jdbc.pool.OracleDataSource;

@Configuration
public class AccountConfig {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

@Bean
public OracleDataSource dataSource() {
    OracleDataSource dataSource = null;
    try {
        dataSource = new OracleDataSource();
        dataSource.setURL("jdbc:oracle:thin:@localhost:1521:xe");
        dataSource.setUser("system");
        dataSource.setPassword("durga");
    } catch (Exception e) {
        e.printStackTrace();
    }
    return dataSource;
}

```

```

@Bean
public AccountService accService() {
    AccountService accService = new AccountServiceImpl();
    return accService;
}

```

```

@Bean
public AccountDao dao() {
    AccountDao dao = new AccountDaoImpl();
    return dao;
}

```

Drawbacks with Autowiring:

1. Autowiring is less exact than normal wiring.
2. Autowiring will not provide configuration metadata to the documentation tools to prepare Documentations.
3. Autowiring will increase confusion when we have more than one bean object of the same type in IOC Container.
4. In Spring applications, if we provide both explicit wiring and autowiring both at a time to a bean object injection then explicit wiring overrides autowiring configurations.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

5. In Spring configuration files, Explicit wiring will improve readability when compared with autowiring.

6. In Spring applications, if we have more and more no of bean objects to inject there it is suggestible to use explicit wiring when compared with autowiring.

7. Autowiring is applicable for only bean objects injection, it is not applicable for simple values injection like primitive values, string values,....

Method Injection

In Spring applications, by default, all the objects are singleton objects provided by IOC Container[ApplicationContext]. In Spring application , as part of dependency injection both container object and contained object are having same scope then application may not get any problem, if container object and contained objects are having different scopes then application may get problem.

EX: In Spring applications, if we inject Course object in Student Object , where if provide Singleton scope to Student object and Prototype scope to Course object then For every request for Student object single Student Object is created , along with single student object single Course object is created with out checking its scope "prototype" which we provided in spring configuration file, it is violating spring scopes rules and regulations.

Student.java

```
public class Student{  
----  
private Course scounrse;  
----  
setXXX()  
getXXX()  
}  
Course.java  
----  
public class Course{  
----  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

spring_beans_config.xml

```
<beans>
<bean id="std" class="com.durgasoft.beans.Student" scope="singleton">
    ---
    <property name="scourse" ref="scourse"/>
</bean>
<bean id="scourse" class="com.durgasoft.beans.Course" scope="prototype">
    ---
</bean>
</beans>
```

Test.java

```
public class Test{
public static void main(String[] args){
ApplicationContext context=new ClasspathXmlApplicationContext("/com/
durgasoft/cfgs/spring_beans_config.xml");
Student std1=(Student)context.getBean("std");
Student std2=(Student)context.getBean("std");
System.out.println(std1); // a111
System.out.println(std2); // a111
System.out.println(std1.getScourse()); // b111
System.out.println(std2.getScourse()); // b111
}
}
```

To overcome the above problem Spring has provided no of solutions, where one of the solution is "Method Injection".

In Spring Framework, Method injection is available in two forms.

1. Lookup Method Injection
2. Arbitrary Method Replacement

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. Lookup Method Injection

In case of Look Method injection, we will declare an abstract class as a factory class and an abstract method as a factory method then we will give an intimation to IOC Container about to generate a sub class for the abstract class and an implementation for the abstract method dynamically.

In this context, IOC Container will prepare dynamic sub classes for abstract factory class and return that objects to the test application as per the requirement.

In Spring Framework, IOC Container will provide dynamic sub classes by using CGLIB third party library which is managed by Spring framework internally.

To give an intimation to the IOC Container about the sub classes generation and implementation for Factory method by providing configuration details in spring configuration file.

To achieve the above requirement, we have to use "<lookup-method>" in beans configuration.

Example:

Account.java

```
package com.durgasoft.beans;

public interface Account {
    public void create();
    public void search();
    public void update();
    public void delete();
}
```

CurrentAccount.java

```
package com.durgasoft.beans;
public class CurrentAccount implements Account {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public void create() {  
    System.out.println("Current Account is Created");  
}  
public void search() {  
    System.out.println("Current Account is Identified");  
}  
public void update() {  
    System.out.println("Current Account is Updated");  
}  
public void delete() {  
    System.out.println("Current Account is Deleted");  
}
```

SavingsAccount.java

```
package com.durgasoft.beans;  
public class SavingsAccount implements Account {  
    public void create() {  
        System.out.println("Savings Account is created");  
    }  
    public void search() {  
        System.out.println("Savings Account is identified");  
    }  
    public void update() {  
        System.out.println("Savings Account is Updated");  
    }  
    public void delete() {  
        System.out.println("Savings Account is Deleted");  
    }  
}
```

AccountFactory.java

```
package com.durgasoft.factory;  
import com.durgasoft.beans.Account;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public abstract class AccountFactory {  
    public abstract Account getAccount();  
}
```

spring_beans_config.xml

```
<beans>  
    <bean id="savingsAccount" class="com.durgasoft.beans.SavingsAccount" />  
    <bean id="currentAccount" class="com.durgasoft.beans.CurrentAccount" />  
    <bean id="savingsAccountFactory"  
        class="com.durgasoft.factory.AccountFactory">  
        <lookup-method name="getAccount" bean="savingsAccount"/>  
    </bean>  
    <bean id="currentAccountFactory" class="com.durgasoft.factory.AccountFactory">  
        <lookup-method name="getAccount" bean="currentAccount"/>  
    </bean>  
</beans>
```

Test.java

```
package com.durgasoft.test;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
import com.durgasoft.beans.CurrentAccount;  
import com.durgasoft.beans.SavingsAccount;  
import com.durgasoft.factory.AccountFactory;  
public class Test {  
    public static void main(String[] args) {  
        ApplicationContext context=new  
        ClassPathXmlApplicationContext("/com/durgasoft/cfgs/spring_beans_config.xml");  
        AccountFactory  
        savingsAccountFactory=(AccountFactory)context.getBean("savingsAccountFactory");  
        SavingsAccount  
        savings_Account=(SavingsAccount)savingsAccountFactory.getAccount();  
        savings_Account.create();  
        savings_Account.search();  
        savings_Account.update();
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
savings_Account.delete();
System.out.println();
```

```
AccountFactory
currentAccountFactory=(AccountFactory)context.getBean("currentAccountFactory");
CurrentAccount
current_Account=(CurrentAccount)currentAccountFactory.getAccount();
current_Account.create();
current_Account.search();
current_Account.update();
current_Account.delete();
}
}
```

Method Replacement:

It is one type of method injection, where we will give an intimation to IOC Container about to replace the existed method implementation with some other method implementation.

To achieve Method Replacement, we have to declare an user defined bean class with a method whose implementation we want to replace and we will provide new implementation for that method by declaring a class and by implementing "MethodReplacer" interface.

MethodReplacer interface contains reimplement(----) method, here we must provide our new implementation in reimplement() method.

Example:

Course.java

```
package com.durgasoft.beans;
public class Course {
    private String cid;
    private String cname;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private int ccost;
setXXX()
getXXX()
public void getCourseDetails(){
    System.out.println("Course Details");
    System.out.println("-----");
    System.out.println("Course Id :"+cid);
    System.out.println("Course Name :"+cname);
    System.out.println("Course Cost :" +ccost);
}
}
```

NewImpl.java

```
package com.durgasoft.beans;
import java.lang.reflect.Method;
import org.springframework.beans.factory.support.MethodReplacer;
public class NewImpl implements MethodReplacer {
public Object reimplement(Object arg0, Method arg1, Object[] arg2) throws Throwable
{
    System.out.println("Course Details");
    System.out.println("-----");
    System.out.println("Course Id :C-222");
    System.out.println("Course Name :.NET");
    System.out.println("Course Cost :20000");
    return null;
}
}
```

spring beans config.xml

```
<beans>
<bean id="newImpl" class="com.durgasoft.beans.NewImpl"/>
<bean id="course" class="com.durgasoft.beans.Course">
    <property name="cid" value="C-111"/>
    <property name="cname" value="Java"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="ccost" value="10000"/>
<replaced-method name="getCourseDetails" replacer="newImpl"/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Course;
public class Test {
public static void main(String[] args) throws Exception {
ApplicationContext context=new
ClassPathXmlApplicationContext("/com/durgasoft/cfgs/spring_beans_config.xml");
Course crs=(Course)context.getBean("course");
crs.getCourseDetails();
}
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Bean Validations in spring

The process of checking data validity before using data in applications is called as Data Validations.

In Web Applications, there are two types of data validations.

- 1.Client Side Data Validations
- 2.Server Side Data Validations

1.Client Side Data Validations: The process of checking whether data is valid or not at client browser before submitting request to the Server is called as Client Side Data Data Validations. To perform client side data validations we will use "Java Script" Functions.

2.Server Side Data Validations: The process of checking whether data is valid or not at server after getting request from client and before using data in application logic is called as Server side data validations. To perform Server side data validations we have to use Java code explicitly.

Note: If we use web Frameworks like Struts, JSF , SPring WEB MVC module then all these frameworks are able to provide their own validations services implicitly to perform data validations.

In Spring Core Module, to perform validations over the data after storing data in Bean objects Spring Framework has provided a predefined interface in the form of "org.springframework.validation.Validator" .

To perform bean validations in Spring applications then we have to use the following steps.

1.Prepare a properties file with all validation messages:

Declare validation messages in the form of key-value pairs, where keys must be validation message code and value must be validation message.

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX: abc.properties

error.uname= User Name is required.
error.password=User Password is required.

2.Prepare Validator class:

The main intention of Validator class is to define all data validation logic w.r.t the bean properties.

Steps:

a)Declare an user defined class.

b)Implement org.springframework.validation.Validator interface.

c)Implement the following Validator interface provided methods.

```
public boolean supports(Class type)
```

```
public void validate(Object obj, Errors errors)
```

Where supports(--) method will check whether the Bean object is supporting data validations or not.

Where validate(---) is able to include data validation logic inorder to perform data validations.

Where "Errors" is able to include all validation messages in the form of Map object inorder to display. To add an error message to Errors object we have to use "rejectValue(----)" method.

d)Declare Resource property and its setXXX() method inorder to represent the name and location of the properties file where all the validation messages are existed.

3)Configure Validator class in spring configuration file:

In spring configuration file, we have to configure Validator class as a bean with Resource property.

4)In test application get all validation messages from MapBindingResults object[Errors] by using getAllErrors() method.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**Employee.java**

```
package com.durgasoft.beans;
public class Employee {
    private String eid;
    private String ename;
    private float esal;
    private int eage;
    private String eemail;
    private String emobile;

    setXXX() and getXXX()

    public void getEmpDetails(){
        System.out.println("Employee Details");
        System.out.println("-----");
        System.out.println("Employee Id :"+eid);
        System.out.println("Employee Name :"+ename);
        System.out.println("Employee Salary :"+esal);
        System.out.println("Employee Email :"+eemail);
        System.out.println("Employee Mobile :"+emobile);
    }
}
```

messages.properties

```
error.eid.empty=Employee Id is required.
error.eid.invalid=Invalid Employee Id.

error.ename.empty=Employee Name is required.

error.esal.invalid=Employee Salary is Invalid.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

error.eage.minage=Employee Age must not be less than 18 years.
error.eage.maxage=Employee Age must not be greater than 30 years.

error.eemail.empty=Employee Email Id is required.
error.eemail.invalid=Employee Email Id is Invalid.

error.emobile.empty=Employee Mobile is required.
error.emobile.invalid=Employee Mobile is Invalid.

EmployeeValidator.java

```
package com.durgasoft.validations;

import com.durgasoft.beans.Employee;
import java.util.Properties;
import org.springframework.core.io.Resource;
import org.springframework.core.io.support.PropertiesLoaderUtils;
import org.springframework.validation.Errors;
import org.springframework.validation.Validator;

public class EmployeeValidator implements Validator{
    private Resource resource;
    public void setResource(Resource resource){
        this.resource=resource;
    }

    @Override
    public boolean supports(Class type) {
        System.out.println("Hello");
        return Employee.class.equals(type);
    }

    @Override
    public void validate(Object obj, Errors errors) {
        try {
            System.out.println("Hello...validate()");
        }
    }
}
```

Checking Class type and Employee.class are equal or not
apan ja class la validation provide karto toch class ahe ki nahi
check karte he method.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

Properties prop = PropertiesLoaderUtils.loadProperties(resource);
Employee emp=(Employee)obj;
if(emp.getEid() == null || emp.getEid() == ""){
    errors.rejectValue("eid", "error.eid.empty", prop.getProperty("error.eid.empty"));
} else{                                key value pair      key
    if(!emp.getEid().startsWith("DSS-")){
        errors.rejectValue("eid", "error.eid.invalid", prop.getProperty("error.eid.invalid"));
    }
}
if(emp.getEname() == null || emp.getEname() == ""){
    errors.rejectValue("ename", "error.ename.empty",
    prop.getProperty("error.ename.empty"));
}
if(emp.getEsal() <= 0.0f){
    errors.rejectValue("esal","error.esal.invalid", prop.getProperty("error.esal.invalid")); }
if(emp.getEage() < 18 ){
    errors.rejectValue("eage", "error.eage.minage",
    prop.getProperty("error.eage.minage")); } else if(emp.getEage() > 30
}{errors.rejectValue("eage", "error.eage.maxage",
prop.getProperty("error.eage.maxage"));
}
if(emp.getEemail() == null || emp.getEemail() == ""){
    errors.rejectValue("eemail","error.eemail.empty",prop.getProperty("error.eemail.empt
y"));
} else{
    if(!emp.getEemail().endsWith("@durgasoft.com")){
        errors.rejectValue("eemail", "error.eemail.invalid",
        prop.getProperty("error.eemail.invalid"));
    }
}
if(emp.getEmobile() == null || emp.getEmobile() == ""){
    errors.rejectValue("emobile", "error.emobile.empty",
    prop.getProperty("error.emobile.empty"));
} else{
    if(!emp.getEmobile().startsWith("91-")){

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
errors.rejectValue("emobile", "error.emobile.invalid",
prop.getProperty("error.emobile.invalid")));
}
}
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

applicationContext.xml

```
<beans>
<bean id="emp" class="com.durgasoft.beans.Employee">
    <property name="eid" value="" />
    <property name="ename" value="" />
    <property name="esal" value="" />
    <property name="eage" value="" />
    <property name="eemail" value="" />
    <property name="emobile" value="" />
</bean>
<bean id="empValidator" class="com.durgasoft.validations.EmployeeValidator">
    <property name="resource"
value="/com/durgasoft/resources/messages.properties"/>
</bean>
</beans>
```

Test.java0

```
package com.durgasoft.test;

import com.durgasoft.beans.Employee;
import com.durgasoft.validations.EmployeeValidator;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.validation.MapBindingResult;
import org.springframework.validation.ObjectError;

public class Test {
    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
        ClassPathXmlApplicationContext("applicationContext.xml");
        Employee emp = (Employee) context.getBean("emp");
        emp.getEmpDetails();
        EmployeeValidator empValidator =
        (EmployeeValidator) context.getBean("empValidator");
        Map<String, String> map = new HashMap<String, String>();
        MapBindingResult results = new MapBindingResult(map,
        "com.durgasoft.beans.Employee");
        empValidator.validate(emp, results);
        List<ObjectError> list = results.getAllErrors();
        for (ObjectError e: list){
            System.out.println(e.getDefaultMessage());
        }
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Event Handling

In general, in GUI Applications, when we click on a button, when select an item in check boxes, radio buttons, List boxes, choice boxes ... automatically an event will be rised, where the generated event will not be handled by the respective GUI components, where the generated events are handled by some component internally called as "Listeners" [Handlers]

In Event Handling, the main intention of Listeners is to listen the events rised by the GUI Components, handle that events and generating results back to the GUI Applications.

Similarly, in spring applications, IOC Container is able to rise events when it was started, refreshed, stopped and closed. In this context, to handle the generated events Spring Framework has provided "Event Handling".

In Spring Framework, Event Handling capability is available with ApplicationContext only, not with BeanFactory. In Spring Framework Event Handling all events are represented in the form of predefined classes in "org.springframework.context.event" package like below.

- 1.ContextRefreshedEvent
- 2.ContextStartedEvent
- 3.ContextStoppedEvent
- 4.ContextClosedEvent
- 5.RequestHandledEvent

Where ContextRefreshedEvent will be rised when we start ApplicationContext or when we access "refresh()" method on ApplicationContext.

Where ContextStartedEvent will be rised when we access "start()" method on ApplicationContext.

Where ContextStoppedEvent will be rised when we access "stop()" method on ApplicationContext.

CONTACT US:**Mobile: +91- 8885 25 26 27****+91- 7207 21 24 27/28****US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

Where ContextClosedEvent will be rised when we access "close()" method on ApplicationContext.

Where RequestHandledEvent will be rised in web applications when request is handled in Spring web applications.

To Handle all the above Events, Spring Framework has provided a Listener in the form of "org.springframework.context.ApplicationListener" interface and it is having the following to execute when the respective event is rised.

```
public void onApplicationEvent(XXXEvent e)
```

Note: ApplicationListener is able to listen all the events by default, if we want to filter the events then we have to provide the respective Event type as Generic parameter.

EX: ApplicationListener<ContextStartedEvent> is able to listen only ContextStartedEvent .

In Spring applications, if we want to implement event handling then we have to use the following steps.

- 1.Create implementation classes for ApplicationListener interface and provide implementation for onApplicationEvent(--) method.
- 2.Configure all implementation classes as bean components in spring configuration file.

Note: To perform Event Handling in spring applications we have to use "org.springframework.context.ConfigurableApplicationContext" container , it is a child interface to ApplicationContext interface.

Example:

ContextRefreshedListenerImpl.java

```
package com.durgasoft.listeners;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.context.ApplicationListener;
import org.springframework.context.event.ContextRefreshedEvent;

public class ContextRefreshedListenerImpl implements
ApplicationListener<ContextRefreshedEvent>{

@Override
public void onApplicationEvent(ContextRefreshedEvent e) {
    System.out.println("Application Context is Refreshed");
}

}
```

ContextStartedListenerImpl.java

```
package com.durgasoft.listeners;

import org.springframework.context.ApplicationListener;
import org.springframework.context.event.ContextStartedEvent;

public class ContextStartedListenerImpl implements
ApplicationListener<ContextStartedEvent>{

@Override
public void onApplicationEvent(ContextStartedEvent e) {
    System.out.println("Application Context Started....");
}

}
```

ContextStoppedListenerImpl.java

```
package com.durgasoft.listeners;

import org.springframework.context.ApplicationListener;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.context.event.ContextStoppedEvent;

public class ContextStoppedListenerImpl implements
ApplicationListener<ContextStoppedEvent>{

    @Override
    public void onApplicationEvent(ContextStoppedEvent e) {
        System.out.println("Application Context Stopped");
    }
}
```

ContextClosedListenerImpl.java

```
package com.durgasoft.listeners;

import org.springframework.context.ApplicationListener;
import org.springframework.context.event.ContextClosedEvent;

public class ContextClosedListenerImpl implements
ApplicationListener<ContextClosedEvent>{

    @Override
    public void onApplicationEvent(ContextClosedEvent e) {
        System.out.println("Application Context has Closed");
    }
}
```

HelloBean.java

```
package com.durgasoft.beans;
public class HelloBean {
    private String uname;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public String getUname() {  
    return uname;  
}  
  
public void setUname(String uname) {  
    this.uname = uname;  
}  
public String wish(){  
    return "Hello "+uname+"!";  
}  
}
```

applicationContext.xml

```
<beans>  
    <bean id="helloBean" class="com.durgasoft.beans.HelloBean">  
        <property name="uname" value="Durga"/>  
    </bean>  
    <bean id="contextRefreshedEvent"  
        class="com.durgasoft.listeners.ContextRefreshedListenerImpl"/>  
    <bean id="contextStartedEvent"  
        class="com.durgasoft.listeners.ContextStartedListenerImpl"/>  
    <bean id="contextStoppedEvent"  
        class="com.durgasoft.listeners.ContextStoppedListenerImpl"/>  
    <bean id="contextClosedEvent"  
        class="com.durgasoft.listeners.ContextClosedListenerImpl"/>  
</beans>
```

Test.java

```
package com.durgasoft.test;  
  
import com.durgasoft.beans.HelloBean;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.ConfigurableApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class Test {  
    public static void main(String[] args) throws Exception {  
        ConfigurableApplicationContext context = new  
        ClassPathXmlApplicationContext("applicationContext.xml");  
        HelloBean hello = (HelloBean) context.getBean("helloBean");  
        System.out.println(hello.wish());  
        context.start();  
        context.refresh();  
        context.stop();  
        context.close();  
    }  
}
```

Custom Events in Spring Applications:

not to read only remembering these things

Custom Events are user defined events which are defined by the developers as per their application requirements.

To manage custom Events in Spring applications we have to use the following steps.

- 1.Create User defined Event class
- 2.Create User defined event Publisher class
- 3.Create Event Handler class
- 4.Configure Event Publisher class and Event Handler class in spring configuration file.
- 5.Create Bean components as per the appl requirements and publish events
- 6.Create Test application and Execute Test application.

1.Create User Event Class:

- a)Declare an user defined class.
- b)Extend org.springframework.context.ApplicationEvent abstract class to user defined class.
- c)Declare public and Object parameterized Constructor and access super class Object parameterized constructor by using "super" keyword.

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

d) Define other methods as per the requirement in Event class.

2.Create User defined event Publisher class

The main intention of event publisher class is to publish the user defined event inorder to handle.

Steps:

- a) Declare an user defined class.
- b) Implement org.springframework.context.ApplicationEventPublisherAware interface in event class.
- c) Provide implementation for setApplicationEventPublisher(--) method inorder to inject ApplicationEventPublisher object.
Note: The main intention to implement ApplicationEventPublisherAware interface is to inject ApplicationEventPublisher object only.
- d) Define a method to publish an event by using the following method from ApplicationEventPublisher .

```
public void publishEvent(ApplicationEvent ae)
```

3.Create User defined Event Handler Class:

The main intention of User defined Event Handler class is to handle the user defined Events.

Steps:

- a) Create an User defined class
- b) Implement org.springframework.context.ApplicationListener interface.
- c) Implement onApplicationEvent(--) Method in user defined class with an application logic.
Note: onApplicationEvent(--) method is able to take the parameter which is specified as generic type to ApplicationListener interface.

Note: By default, Listeners are able to handle all the Listeners, but, if we want to filter the Listeners then we have to use Generic type to ApplicationListener.

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: In Spring Event Handling, by default, ApplicationContext is able to handle the events synchronously, but, if we want to handle the events Asynchronously then we have to use ApplicationEventMustcaster interface.

Note: Spring 4.2 version has provided very good annotations support for Event Handling in the form of the following Annotations.

1) @EventListener({Event1.class, Event2.class...})

Where Event1.class, Event2.class,... are Event class types which we want to process.

EX: @EventListener({ContextRefreshedEvent.class, ContextStoppedEvent.class})

2) @Async() : It will be used to process events asynchronously.

4) Prepare Bean components and publish events:

In the application, as per the requirement we are able to publish the events by using publishEvent(--) method.

AccountEvent.java

```
package com.durgasoft.events;
```

```
import java.io.FileOutputStream;
import java.util.Date;
import org.springframework.context.ApplicationEvent;

public class AccountEvent extends ApplicationEvent{
    static FileOutputStream fos;
    static{
        try {
            fos=new FileOutputStream("E:/logs/log.txt", true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    private String message;
    public AccountEvent(Object obj, String message) {
        super(obj);
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
    this.message=message;
}

public void generateLog(){
    //System.out.println("*****"+message+"*****");
    try {
        message=new Date().toString()+":"+message;
        message=message+"\n";
        byte[] b=message.getBytes();
        fos.write(b);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}
```

AccountEventPublisher.java

```
package com.durgasoft.events;

import org.springframework.context.ApplicationEventPublisher;
import org.springframework.context.ApplicationEventPublisherAware;

public class AccountEventPublisher implements ApplicationEventPublisherAware{
    private ApplicationEventPublisher publisher;
    @Override
    public void setApplicationEventPublisher(ApplicationEventPublisher publisher) {
        this.publisher=publisher;
    }
    public void publish(String message){
        AccountEvent ae=new AccountEvent(this, message);
        publisher.publishEvent(ae);
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

AccountEventHandler.java

```
package com.durgasoft.events;

import org.springframework.context.ApplicationListener;

public class AccountEventHandler implements ApplicationListener<AccountEvent> {

    @Override
    public void onApplicationEvent(AccountEvent e) {
        e.generateLog();
    }
}
```

Account.java

```
package com.durgasoft.beans;

import com.durgasoft.events.AccountEventPublisher;

public class Account {
    private AccountEventPublisher publisher;
    public void setPublisher(AccountEventPublisher publisher){
        this.publisher=publisher;
    }
    public void createAccount(){
        System.out.println("Account Created");
        publisher.publish("AccountCreated");
    }
    public void searchAccount(){
        System.out.println("Account Identified");
        publisher.publish("AccountIdentified");
    }
    public void updateAccount(){
        System.out.println("Account Updated");
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
    publisher.publish("AccountUpdated");
}
public void deleteAccount(){
    System.out.println("Account Deleted");
    publisher.publish("AccountDeleted");
}
}
```

applicationContext.xml

```
<beans>
    <bean id="account" class="com.durgasoft.beans.Account">
        <property name="publisher" ref="accountEventPublisher"/>
    </bean>
    <bean id="accountEventHandler"
        class="com.durgasoft.events.AccountEventHandler"/>
    <bean id="accountEventPublisher"
        class="com.durgasoft.events.AccountEventPublisher"/>
</beans>
```

Test.java

```
package com.durgasoft.test;
import com.durgasoft.beans.Account;
import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class Test {
    public static void main(String[] args) throws Exception {
        ConfigurableApplicationContext context=new
        ClassPathXmlApplicationContext("applicationContext.xml");
        Account account=(Account)context.getBean("account");
        account.createAccount();
        account.searchAccount();
        account.updateAccount();
        account.deleteAccount();
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Internationalization in SPRING

Designing java applications w.r.t Local Users is called as Internationalization.

To provide Internationalization services to the users, first, we have to devide all the users into groups as per locality, for this, we have to use the following parameters.

1.language: It able to represent two lower case letters.

EX: en, it, hi,

2.country: It will be represented in the form of two Upper case letters.

EX: US, IN, IT,.....

3.System Variant[OS]: It will be represented in the form of three lower case letters.

EX: win, uni, lin,.....

In java applications, to represent a group of local users JAVA has provided a predefined class in the form of "java.util.Locale".

To create Locale class object we have to use the following Constructors.

```
public Locale(String lang)
public Locale(String lang, String country)
public Locale(String lang, String country, String sys_Variant)
```

EX:

```
Locale l1=new Locale("en");
Locale l2=new Locale("en", "US");
Locale l3=new Locale("en", "US", "win");
```

In Java applications, we are able to provide the following services as part of Internationalization.

1.Number Formations

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 2.Date Formations
- 3.Message Formations

1. Number Formations:

It can be used to represent a number w.r.t a particular Locale. It will use java.text.NumberFormat class to represent a number.

Steps:

- a)Create Locale object.
- b)Create NumberFormat class object by using getInstance() Factory method.
- c)Represent Number as per the Locale by using format(-) method.

2.Date Formations:

It can be used to represent a Date w.r.t a particular Locale, for this, it will use java.text.DateFormat class.

Steps:

- a)Create Locale object.
- b)Create DateFormat class object by using getDateInstance(--) Factory method.
- c)Represent Date w.r.t the Locale by using format(--) method.

3.Message Formations:

It can be used to represent messages w.r.t a particular Locale, for this, we have to use properties files and java.util.ResourceBundle class.

Steps:

- a)Create properties files with all the messages in the form of key-value pairs.
Note: properties files names must be provided in the following format.
baseName_lang_country.properties
- b)Create ResourceBundle object by using getBundle(--) Factory method.
- c)Get Message from ResourceBundle object by using getString(-) method.

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:`com/durgasoft/resources/abc_en_US.properties`

welcome = Welcome To en US Users.

`com/durgasoft/resources/abc_it_IT.properties`

welcome = Welcome To it IT Users.

Test.java

```
package com.durgasoft;
import java.text.DateFormat;
import java.text.NumberFormat;
import java.util.Date;
import java.util.Locale;
import java.util.ResourceBundle;
public class Test {
    public static void main(String[] args) throws Exception {
        Locale l = new Locale("it", "IT");
        NumberFormat num_Format = NumberFormat.getInstance(l);
        System.out.println(num_Format.format(1234567.23456));
        DateFormat date_Format = DateFormat.getDateInstance(0, l);
        System.out.println(date_Format.format(new Date()));
        ResourceBundle resource_Bundle =
        ResourceBundle.getBundle("com/durgasoft/resources/abc", l);
        System.out.println(resource_Bundle.getString("welcome"));
    }
}
```

To provide Message formations in Spring applications, Spring has provided a predefined interface in the form of "org.springframework.context.MessageSource" .

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

For MessageSource interface Spring Framework has provided the following two implementation classes.

org.springframework.context.support.ResourceBundleMessageSource
org.springframework.context.support.ReloadableResourceBundleMessageSource

Where ResourceBundleMessageSource is able to get messages from properties files on the basis of the provided locale.

Where ReloadableResourceBundleMessageSource is able to get messages from both properties files and from XML files.

Steps:

1. Declare properties files with the messages and with the following format for properties files names.

baseName_lang_Country.properties.

2. Declare a Bean class with MessageSource type property and the respective setter method and the required business methods.

Note: To get a message from MessageSource object we have to use the following method.

`public String getMessage(String key, Object[] place_holder_values, Locale l)`

Where "key" is key of the message defined in properties file.

Where "Object[]" must be provided to provide values to the place holders which we defined in messages in properties files, if place holders are not existed in messages then we have to provide "null" value as Object[].

Where Locale is able to represent the constants like US, FRANCE, IN,... from Locale class inorder to recognize the properties file.

3. Configure bean class in properties file and inject either ResourceBundleMessageSource or ReloadableResourceBundleMessageSource object as reference in Bean object and provide base name as property for MessageSource object.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4.In Main class, in main(), get Bean object and access business method.

Example-1:

abc_en_US.properties

welcome = Welcome To {0} and {1} User.

abc_fr_FR.properties

welcome = Welcome To {0} and {1} Users.

I18NBean.java

package com.durgasoft.beans;

import java.util.Locale;

import org.springframework.context.MessageSource;

public class I18NBean {

 private MessageSource messageSource;

 public void setMessageSource(MessageSource messageSource) {

 this.messageSource = messageSource;

 }

 public void displayMessage(){

 System.out.println("Message :" +messageSource.getMessage("welcome", new Object[]{"fr", "FRANCE"}, Locale.FRANCE));

 System.out.println("Message :" +messageSource.getMessage("welcome", new Object[]{"en", "US"}, Locale.US));

 }

}

applicationContext.xml

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<beans>
<bean id="i18nBean" class="com.durgasoft.beans.I18NBean">
    <property name="messageSource" ref="resourceBundleMessageSource"/>
</bean>
<bean id="resourceBundleMessageSource"
class="org.springframework.context.support.ResourceBundleMessageSource">
    <property name="basename" value="com/durgasoft/resources/abc"/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;

import com.durgasoft.beans.I18NBean;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Test {
    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        I18NBean bean = (I18NBean)context.getBean("i18nBean");
        bean.displayMessage();
    }
}
```

If we want to take messages from xml file by using Reloadable ResourceBundleMessageSource then we have to use define xml files with the name like `baseName_lang_Country.xml` and with the following tags to represent messages.

```
<properties>
<entry key="message_Key"> Message_Value </entry>
-----
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
-----  
</properties>
```

In XML files we must provide the following DTD definition.

```
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
```

Example:

abc_en_US.xml

```
-----  
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">  
<properties>  
    <entry key="welcome"> Welcome to en US User from XML </entry>  
</properties>
```

abc_fr_FR.xml

```
-----  
<?xml version="1.0" encoding="UTF-8"?>  
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">  
<properties>  
    <entry key="welcome"> Welcome to fr France User from XML </entry>  
</properties>
```

spring_beans_config.xml

```
-----  
<beans>  
    <bean id="i18nBean" class="com.durgasoft.beans.I18NBean">  
        <property name="messageSource"  
        ref="reloadableResourceBundleMessageSource"/>  
    </bean>  
    <bean id="reloadableResourceBundleMessageSource"  
    class="org.springframework.context.support.ReloadableResourceBundleMessageSourc  
e">  
        <property name="basename" value="com/durgasoft/resources_xml/abc"/>  
    </bean>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

</beans>

I18NBean.java

```
-----  
package com.durgasoft.beans;  
import java.util.Locale;  
import org.springframework.context.MessageSource;  
  
public class I18NBean {  
    private MessageSource messageSource;  
  
    public void setMessageSource(MessageSource messageSource) {  
        this.messageSource = messageSource;  
    }  
    public void displayMessage(){  
        System.out.println("Message :" +messageSource.getMessage("welcome", null,  
Locale.FRANCE));  
        System.out.println("Message :" +messageSource.getMessage("welcome", null,  
Locale.US));  
    }  
}
```

Test.java

```
package com.durgasoft.test;  
import com.durgasoft.beans.I18NBean;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
public class Test {  
    public static void main(String[] args)throws Exception {  
        ApplicationContext context=new  
ClassPathXmlApplicationContext("applicationContext.xml");  
        I18NBean bean = (I18NBean)context.getBean("i18nBean");  
        bean.displayMessage();  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Bean Manipulations and Bean Wrappers

In Bean Manipulation, we are able to perform the following actions.

- 1.Getting Beans Information explicitly like properties and their setXXX() and getXXX() methods.
- 2.Creating JavaBean Objects, checking bean property types, copying bean properties, etc.
- 3.Accessing fields without standard getters and setters.
- 4.analyze and manipulate standard JavaBeans like to get and set property values, get property descriptors, and query the readability/writability of properties and setting of index properties.

In Java, we are able to get beans descriptions like bean properties information like their names and the corresponding setXXX() and getXXX() methods information by using "Beans Introspection".

If we want to get beans data explicitly then we have to java.beans.BeanInfo interface, to get BeanInfo object then we have to use the following method from java.beans.Introspector class.

```
public BeanInfo getBeanInfo(Class bean_class_type)  
EX: BeanInfo beanInfo = Introspector.getBeanInfo(MyBean.class);
```

To get All properties information of the Bean object we have to use "java.beans.PropertyDescriptor" class. To get all properties description in the form of PropertyDescriptor objects in an array then we have to use the following method from BeanInfo .

```
public PropertyDescriptor[] getPropertyDescriptors()  
EX: PropertyDescriptor[] props = beanInfo.getPropertyDescriptors();
```

Example:

```
package com.durgasoft.core;  
public class Employee {
```

Bean Manipulation helps you control, modify, or enhance beans before they are used.

BeanWrapper lets you easily set or get bean properties dynamically.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private int eno;  
private String ename;  
private float esal;  
private String eaddr;  
  
setXXX() and getXXX()  
}
```

Test.java

```
package com.durgasoft.core;  
  
import java.beans.BeanInfo;  
import java.beans.Introspector;  
import java.beans.PropertyDescriptor;  
  
public class Test {  
    public static void main(String[] args) throws Exception {  
        BeanInfo beanInfo = Introspector.getBeanInfo(Employee.class);  
        PropertyDescriptor[] property_desc = beanInfo.getPropertyDescriptors();  
        for(PropertyDescriptor p: property_desc){  
            System.out.println(p);  
        }  
        MethodDescriptor[] meths = beanInfo.getMethodDescriptors();  
        for(MethodDescriptor m: meths){  
            System.out.println(m.getName());  
        }  
    }  
}
```

In Spring framework, to create beans and to manipulate beans explicitly Spring Framework has provided predefined library in the form of "org.springframework.beans".

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In spring "org.springframework.beans" package has provided the following classes and interfaces to perform manipulations on beans.

BeanInfoFactory: It is an alternative to "Beans Introspection" provided by Spring Framework, it can be used to get details about the Bean objects like properties details, events details,... by using java.beans.BeanInfo object internally . Spring Framework has provided a seperate predefined implementation class for BeanInfoFactory interface in the form of "org.springframework.beans.ExtendedBeanInfoFactory" class.

To get BeanInfo object we have to use the following method from BeanInfoFactory interface.

```
public BeanInfo getBeanInfo(Class bean_Class_Type)
```

Note: BeanInfoFactory implementation, org.springframework.beans.ExtendedBeanInfoFactory, accepts JavaBeans "non-standard" setter methods as 'writable' which returns some values instead of void.

Example:

Employee.java

```
package com.durgasoft.beans;
public class Employee {
    private int eno;
    private String ename;
    private float esal;
    private String eaddr;

    public int getEno() {
        return eno;
    }

    public int setEno(int eno) {
        this.eno = eno;
        return eno;
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftwareonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
}

public String getEname() {
    return ename;
}

public void setEname(String ename) {
    this.ename = ename;
}

public float getEsal() {
    return esal;
}

public void setEsal(float esal) {
    this.esal = esal;
}

public String getEaddr() {
    return eaddr;
}

public void setEaddr(String eaddr) {
    this.eaddr = eaddr;
}

}

Test.java

package com.durgasoft.test;

import com.durgasoft.beans.Employee;
import java.beans.BeanInfo;
import java.beans.PropertyDescriptor;
import org.springframework.beans.BeanInfoFactory;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.beans.ExtendedBeanInfoFactory;
```

```
public class Test {  
    public static void main(String[] args) throws Exception {  
        BeanInfoFactory factory = new ExtendedBeanInfoFactory();  
        BeanInfo bean_Info = factory.getBeanInfo(Employee.class);  
        System.out.println(bean_Info);  
        PropertyDescriptor[] props = bean_Info.getPropertyDescriptors();  
        for(PropertyDescriptor p: props){  
            System.out.println(p);  
        }  
        MethodDescriptor[] meths = beanInfo.getMethodDescriptors();  
        for(MethodDescriptor m: meths){  
            System.out.println(m.getName());  
        }  
    }  
}
```

1.BeanWrapper: BeanWrapper provides methods to create Bean objects explicitly , to analyze and manipulate standard JavaBeans like the ability to get and set property values, get property descriptors and checks the readability/writability of properties. BeanWrapper is also supports setting of index properties.

Spring Framework has provided a separate predefined implementation class for BeanWrapper in the form of "BeanWrapperImpl".

To set values to the Bean object through Bean Wrapper class we have to use the following method from BeanWrapper class.

```
public void setPropertyValue(String prop_Name, Object value)
```

Note: If we want to set all the properties at a time to Bean object, first, we have to set property names and their values in the form of Map object then set that Map object to BeanWrapper object, for this, we have to use the following method.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public void setPropertyValues(Map map)
```

To get property value explicitly from Bean object we have to use the following method from BeanWrapper class.

```
public Object getProperty(String name)
```

To get Bean object explicitly through BenWrapper we have to use the following method from BeNWrapper.

```
public Object getWrappedInstance()
```

To copy the properties values from one Bean object to another Bean object we have to use the following method from "org.springframework.beans.BeanUtils" class.

```
public void copyProperties(Object source, Object target)
```

Where Source object and target objects may be the objects of Same class or different classes having same property names and same property data types.

To Check whether the property is readable or writable then we have to use the following methods from BeanWrapper class.

```
public boolean isReadableProperty(String prop_Name)
```

```
public boolean isWritableProperty(String prop_Name)
```

Example:

Employee.java

```
package com.durgasoft.beans;
public class Employee {
    private int eno;
    private String ename;
    private float esal;
    private String eaddr;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public int getEno() {  
    return eno;  
}  
  
public void setEno(int eno) {  
    this.eno = eno;  
}  
  
public String getEname() {  
    return ename;  
}  
  
public void setEname(String ename) {  
    this.ename = ename;  
}  
  
public float getEsal() {  
    return esal;  
}  
  
public void setEsal(float esal) {  
    this.esal = esal;  
}  
  
public String getEaddr() {  
    return eaddr;  
}  
  
public void setEaddr(String eaddr) {  
    this.eaddr = eaddr;  
}  
  
public void displayEmpDetails(){  
    System.out.println("Employee Details");  
    System.out.println("-----");  
    System.out.println("Employee Id : "+eno);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
System.out.println("Employee Name :" +ename);
System.out.println("Employee Salary :" +esal);
System.out.println("Employee Address:" +eaddr);

}
```

Test1.java

```
package com.durgasoft.test;

import com.durgasoft.beans.Employee;
import org.springframework.beans.BeanWrapper;
import org.springframework.beans.BeanWrapperImpl;

public class Test1 {
    public static void main(String[] args) throws Exception {

        BeanWrapper bw = new BeanWrapperImpl(Employee.class);
        bw.setPropertyValue("eno", 111);
        bw.setPropertyValue("ename", "AAA");
        bw.setPropertyValue("esal", 5000.0f);
        bw.setPropertyValue("eaddr", "Hyd");

        Employee emp = (Employee) bw.getWrappedInstance();
        System.out.println(emp);
        emp.displayEmpDetails();
        System.out.println();

        Map<String, String> map = new HashMap<String, String>();
        map.put("eno", "222");
        map.put("ename", "BBB");
        map.put("esal", "6000");
        map.put("eaddr", "Hyd");
        bw.setPropertyValues(map);
        System.out.println(emp);
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
emp.displayEmpDetails();
```

```
System.out.println("Employee Details");
System.out.println("-----");
System.out.println("Employee No :"+bw.getPropertyValue("eno"));
System.out.println("Employee Name :"+bw.getPropertyValue("ename"));
System.out.println("Employee Salary :"+bw.getPropertyValue("esal"));
System.out.println("Employee Address :"+bw.getPropertyValue("eaddr"));
}
}
```

Test2.java

```
package com.durgasoft.test;
import com.durgasoft.beans.Employee;
import org.springframework.beans.BeanUtils;
import org.springframework.beans.BeanWrapper;
import org.springframework.beans.BeanWrapperImpl;

public class Test2 {
    public static void main(String[] args) throws Exception {
        Employee emp1 = new Employee();
        BeanWrapper bw = new BeanWrapperImpl(emp1);
        bw.setPropertyValue("eno", 111);
        bw.setPropertyValue("ename", "AAA");
        bw.setPropertyValue("esal", 5000.0f);
        bw.setPropertyValue("eaddr", "Hyd");
        System.out.println(emp1);
        emp1.displayEmpDetails();

        Employee emp2 = new Employee();
        BeanUtils.copyProperties(emp1, emp2);
        System.out.println(emp2);
        emp2.displayEmpDetails();
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Property Editors

The main intention of the PropertyEditors is to convert data from text to Object and from Object to text. Property Editors is used to convert xml text data into object of our bean .

In general, in J2SE, in Java Beans, PropertyEditor was originally designed to be used in Swing applications. JavaBeans specification defines API to introspect and extract the bean inner details which can be used to show bean properties visually as components and edit them by using PropertyEditors in Build Tools.

In Spring Applications, we will provide all values in spring configuration file as text values , but, Spring framework has to store these text values into the bean objects as the objects like Byte, Integer, String, Long..... In this context, to convert data from textual rep-resentation to the respective objects Spring framework will use a feature "Property Editors".

To convert data from text form to Objects , Spring Framework has provided the following Predefined Property Editors.

1.ByteArrayPropertyEditor: Editor for byte arrays. Strings will simply be converted to their corresponding byte representations.

it is used to convert text data into byteArray format .

2.ClassEditor: Parses Strings representing classes to actual classes .

it is used to convert text data to string classes actual classes.

3.CustomBooleanEditor: Customizable property editor for Boolean properties.

it is used to convert text data to boolean format.

4.CustomCollectionEditor: Property editor for Collections, converting any source Collection to a given target Collection type. Custom Date Editor Customizable property editor for java.util.Date, supporting a custom Date Format.

5.CustomNumberEditor: Customizable property editor for any Number subclass like Integer, Long, Float, Double.

6.FileEditor: Capable of resolving Strings to java.io.File objects.

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

7.InputStreamEditor: One-way property editor, capable of taking a text string and producing (via an intermediate ResourceEditor and Resource) an InputStream, so InputStream properties may be directly set as Strings.

8.LocaleEditor: Capable of resolving Strings to Locale objects and vice versa (the String format is [country][variant], which is the same thing the `toString()` method of Locale provides).

9.PatternEditor: Capable of resolving Strings to `java.util.regex.Pattern` objects and vice versa.

10.PropertiesEditor: Capable of converting Strings (formatted using the format as defined in the javadocs of the `java.util.Properties` class) to Properties objects.

11.StringTrimmerEditor: Property editor that trims Strings. Optionally allows transforming an empty string into a null value.

12.URLEditor: Capable of resolving a String representation of a URL to an actual URL object.

Spring Framework has provided an approach to provide custom Property Editors, for this , we have to use the following steps.

- 1.Create User defined Property Editor class by extending `java.beans.PropertyEditorSupport` class.
- 2.Override `setAsText(...)` method in user defined Property Editor.
- 3.Configure `org.springframework.beans.factory.config.CustomEditorConfigurer` in spring configuration file with the property "customEditors" of Map type with a key-value pair, where key is the class type for which the property editor is defined and value is the custom property editor.
- 4.Prepare Spring application as it is .

Example:**EmployeeAddress.java****CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
package com.durgasoft.beans;
public class EmployeeAddress {
    private String pno;
    private String street;
    private String city;
    private String country;

    public String getPno() {
        return pno;
    }

    public void setPno(String pno) {
        this.pno = pno;
    }

    public String getStreet() {
        return street;
    }

    public void setStreet(String street) {
        this.street = street;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

    public String getCountry() {
        return country;
    }

    public void setCountry(String country) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
    this.country = country;  
}  
  
}
```

Employee.java

```
package com.durgasoft.beans;  
public class Employee {  
    private String eid;  
    private String ename;  
    private float esal;  
    private EmployeeAddress eaddr;  
  
    public String getEid() {  
        return eid;  
    }  
  
    public void setEid(String eid) {  
        this.eid = eid;  
    }  
  
    public String getEname() {  
        return ename;  
    }  
  
    public void setEname(String ename) {  
        this.ename = ename;  
    }  
  
    public float getEsal() {  
        return esal;  
    }  
  
    public void setEsal(float esal) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
this.esal = esal;  
}  
  
public EmployeeAddress getEaddr() {  
    return eaddr;  
}  
  
public void setEaddr(EmployeeAddress eaddr) {  
    this.eaddr = eaddr;  
}  
  
public void getEmpDetails(){  
    System.out.println("Employee Details");  
    System.out.println("-----");  
    System.out.println("Employee Id : "+eid);  
    System.out.println("Employee Name : "+ename);  
    System.out.println("Employee Salary : "+esal);  
    System.out.println("Employee Address Details:");  
    System.out.println("-----");  
    System.out.println("PNO : "+eaddr.getPno());  
    System.out.println("STREET : "+eaddr.getStreet());  
    System.out.println("CITY : "+eaddr.getCity());  
    System.out.println("COUNTRY : "+eaddr.getCountry());  
}  
}  
  
EmployeeAddressEditor.java  
  
package com.durgasoftware.beans;  
  
import java.beans.PropertyEditorSupport;  
  
public class EmployeeAddressEditor extends PropertyEditorSupport{  
  
    @Override  
    public void setAsText(String text) throws IllegalArgumentException {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftwareonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
String[] str = text.split("-");  
System.out.println(text);  
EmployeeAddress eaddr = new EmployeeAddress();  
eaddr.setPno(str[0]);  
eaddr.setStreet(str[1]);  
eaddr.setCity(str[2]);  
eaddr.setCountry(str[3]);  
super.setValue(eaddr);  
  
}  
}  
}
```

Test.java

```
package com.durgasoft.test;  
  
import com.durgasoft.beans.Employee;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
public class Test {  
    public static void main(String[] args) throws Exception {  
        ApplicationContext context = new  
ClassPathXmlApplicationContext("/com/durgasoft/cfgs/spring_beans_config.xml");  
        Employee emp = (Employee) context.getBean("emp");  
        emp.getEmpDetails();  
    }  
}
```

spring_beans_config.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean id="emp" class="com.durgasoft.beans.Employee">
    <property name="eid" value="E-111"/>
    <property name="ename" value="Durga"/>
    <property name="esal" value="50000"/>
    <property name="eaddr" value="23/3rt-M G Road-Hyd-India"/>

</bean>
<bean class="org.springframework.beans.factory.config.CustomEditorConfigurer">
    <property name="customEditors">
        <map>
            <entry key="com.durgasoft.beans.EmployeeAddress"
                  value="com.durgasoft.beans.EmployeeAddressEditor"/>
        </map>
    </property>
</bean>
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

PROFILING

Profiling in Spring means switching between different configurations based on the environment (e.g., development, testing, production).

In general, in project lifecycle, we have to perform development, testing, production mainly . At each and every phase of project lifecycle we may use databases, debugging tools, testing tools, with different configuration details.

In general, in all project lifecycle phases we will provide the required configuration details manually, it may increase problems to the applications , in this context, Spring3.x version has provided an automated solution inorder to provide the corresponding configuration details wrt the lifecycle phases, for this, Spring framework has provided "Profiling" feature.

Note: IN Project Implementation , we may use database configuration details like datasource names, connection pool names, JNDI names in Server,.....

If we want to implement Profiling in Spring applications then we have to use the following steps.

1.Create a seperate spring configuration file for each and every phase of the project lifecycle with the respective configuration details.

EX: spring-context-development.xml
spring-context-testing.xml
spring-context-production.xml

Note: spring configuration XML File format must be <FileName>-phase_Name.xml

2.In all Spring Configuration files we must provide "profile" attribute in <beans> tag with the respective lifecycle phyase name.

EX: spring-context-development.xml

```
-----  
<beans profile="development">  
-----  
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX: spring-context-production.xml

```
<beans profile="production">  
-----  
</beans>
```

3. Provide project lifecycle phase in System property with the key "spring.profiles.active" in Main Application.

EX: System.setProperty("spring.profiles.active", "development");

4. In main Application, we have to use GenericXmlApplicationContext as Container and load all the spring configuration files with ctx.load(--,--); method and rdfresh context.

EX:

```
GenericXmlApplicationContext context = new GenericXmlApplicationContext();  
context.load("spring-context-development.xml", "spring-context-production.xml");  
context.refresh();
```

Example:

AccountBean.java

```
package com.durgasoft.beans;  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.ResultSet;  
import java.sql.ResultSetMetaData;  
import java.sql.Statement;  
  
public class AccountBean {  
    private String driverClass;  
    private String driverURL;  
    private String dbUserName;  
    private String dbPassword;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

`setXXX()` and `getXXX()`

```

public void listAccounts() {
    try {
        Class.forName(driverClass);
        Connection con = DriverManager.getConnection(driverURL,
dbUserName, dbPassword);
        Statement st = con.createStatement();
        ResultSet rs = st.executeQuery("select * from account");
        ResultSetMetaData md = rs.getMetaData();
        int columns = md.getColumnCount();
        for(int i=1; i<= columns; i++) {
            System.out.print(md.getColumnName(i)+"\t");
        }
        System.out.println();
        System.out.println("-----");
        while(rs.next()) {
            for(int i=1; i<=columns; i++) {
                System.out.print(rs.getString(i)+"\t");
            }
            System.out.println();
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

applicationContext-development.xml

```

<beans .... profile="development">
    <bean id="accBean" class="com.durgasoft.beans.AccountBean">
        <property name="driverClass" value="oracle.jdbc.OracleDriver"/>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="driverURL"
value="jdbc:oracle:thin:@localhost:1521:xe"/>
    <property name="dbUserName" value="system"/>
    <property name="dbPassword" value="durga"/>
</bean>

</beans>
```

applicationContext-production.xml

```
<beans .... profile="production">
    <bean id="accBean" class="com.durgasoft.beans.AccountBean">
        <property name="driverClass" value="com.mysql.jdbc.Driver"/>
        <property name="driverURL"
value="jdbc:mysql://localhost:3306/durgadb"/>
        <property name="dbUserName" value="root"/>
        <property name="dbPassword" value="root"/>
    </bean>
</beans>
```

Test.java

```
package com.durgasoft.test;

import org.springframework.context.support.GenericXmlApplicationContext;
import com.durgasoft.beans.AccountBean;

public class Test {

    public static void main(String[] args) throws Exception {
        System.setProperty("spring.profiles.active", "production");
        GenericXmlApplicationContext context = new
        GenericXmlApplicationContext();
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
    context.load("applicationContext-development.xml", "applicationContext-  
production.xml");  
    context.refresh();  
    AccountBean accBean = (AccountBean)context.getBean("accBean");  
    accBean.listAccounts();  
  
}  
}
```

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring Expression Language [SpEL]

Expression Language is a programming language, it will provide simplified syntaxes to manipulate Objects and their properties.

EX:

1.JSP EL: To evaluate the objects and their properties like request, session, application,... and their parameters and attributes.

2.Struts2.x OGNL: To evaluate the objects and their properties like Value Stack, CentralContext, request, application.....

3.JBOSS EL: To evaluate Objects and their properties which are related to the JBOSS implementations.

4.SpEL: To evaluate bean objects and their properties in SPring applications

SpEL: It is an Expression Language, it has provided simplified syntaxes to manipulate objects and their properties during Runtime of the applications

To prepare and Evaluate Expressions in SpEL, Spring has provided very good Predefined Library in the form of "org.springframework.exprssion" package.

In SPring applications, if we want to prepare and evaluate expressions we have to use the following steps.

1.Create ExpressionParser object:

ExpressionParser is able to manage expressions and it able to have expression evaluation mechanisms.

To represent Expression Parser Spring Framework has provided a predefined interface in the form of org.springframework.expression.ExpressionParser .

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

For ExpressionParser interface , Spring framework has provided a predefined implementation class in the form of "org.springframework.expression.spel.standard.SpELExpressionParser" .

EX: ExpressionParser parser = new SpELExpressionParser();

Note: ExpressinParser is able to evaluate the expressions by using StandardEvaluationContext, it able to evaluate the expressions against objects by preparing Object Graphs internally.

2.Create Expression object:

In SpEL, Expression object is able to represent single Expression. To represent Expression , SpEL has provided a predefined interface in the form of "org.springframework.expression.Expression" . For Expression interface SpEL has provided a predefined implementation class in the form of "org.springframework.expression.spel.standard.SpELExpression" .

To prepare expression and to get Expression object we have to use the following method from ExpressionParser .

public Expression parseExpression(String expression)
EX: Expression expr = parser.parseExpression("10+10");

3.Get Result of the Expression Evaluation:

To get expression result we have to use the following method from Expression.

public Object getValue()

EX: int val = (Object) expr.getValue();

EX:

package com.durgasoft.test;

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.expression.Expression;
import org.springframework.expression.ExpressionParser;
import org.springframework.expression.spel.standard.SpelExpressionParser;

public class Test {

    public static void main(String[] args) throws Exception {
        ExpressionParser parser = new SpelExpressionParser();
        Expression expr = parser.parseExpression("10+10");
        int val1 = (Integer) expr.getValue();
        System.out.println(val1);

        expr = parser.parseExpression("10*10");
        int val2 = (Integer) expr.getValue();
        System.out.println(val2);

        expr = parser.parseExpression("'"abc'+def'");
        String val3 = (String) expr.getValue();
        System.out.println(val3);
    }
}
```

Example with StandardEvaluationContext:

CalculatorBean.java

```
package com.durgasoft.beans;

public class CalculatorBean {
    private int num1;
    private int num2;

    setXXX() and getXXX()

    public int add() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        return num1+num2;
    }
    public int sub() {
        return num1-num2;
    }
    public int mul() {
        return num1*num2;
    }
}
```

Test.java

```
package com.durgasoft.test;

import org.springframework.expression.Expression;
import org.springframework.expression.ExpressionParser;
import org.springframework.expression.spel.standard.SpelExpressionParser;
import org.springframework.expression.spel.support.StandardEvaluationContext;

import com.durgasoft.beans.CalculatorBean;

public class Test {

    public static void main(String[] args) {
        CalculatorBean cal = new CalculatorBean();
        StandardEvaluationContext context = new
StandardEvaluationContext(cal);
        ExpressionParser parser = new SpelExpressionParser();

        Expression expr1 = parser.parseExpression("num1");
        expr1.setValue(context, "10");

        Expression expr2 = parser.parseExpression("num2");
        expr2.setValue(context, "5");
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        System.out.println("num1 :"+cal.getNum1());
        System.out.println("num2 :"+cal.getNum2());
        System.out.println("ADD :" +cal.add());
        System.out.println("SUB :" +cal.sub());
        System.out.println("MUL :" +cal.mul());
    }
}
```

SpEL Features:

- 1.Expressions
- 2.Operators
- 3.Variables
- 4.Method Invocations
- 5.Collections

1.Expressions

SpEL is able to allow two types of Expressions.

- 1.Literal Expressions
- 2.Regular Expressions

1.Literal Expressions

---> It able to allow only literals inside the Expressions.

2.Regular Expressions

---> It able to check the specified String against the provided Regular Expressions, If the provided String is as per the provided Regular Expression then it will return true value otherwise it will return false value.

To compare String with Regular Expression we have to use "matches" operator, it is a boolean operator, it will check whether the provided String is satisfying the provided Regular Expression or not.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Syntax:

'String_Data' matches 'Reg_Expression'

Example:

```
package com.durgasoft.test;
import org.springframework.expression.Expression;
import org.springframework.expression.ExpressionParser;
import org.springframework.expression.spel.standard.SpelExpressionParser;
public class Test {

    public static void main(String[] args) throws Exception {
        ExpressionParser parser = new SpelExpressionParser();
        Expression expr = parser.parseExpression("10 + 10");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("'abc' + 'def'");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("0xABCD*10+6");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("'Spring' matches 'Sp.*'");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("'abc@durgasoft.com' matches '[a-
z]*@durgasoft.com']");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("'abc@gmail.com' matches '[a-
z]*@durgasoft.com']");
        System.out.println(expr.getValue());
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2.Operators:

In SpEL, to prepare Expressions we are able to use the following operators.

i).Arithmetic Operators:

+, -, *, /, %,.....

ii).Logical Operators:

and or &&
or or ||
not or !

iii).Comparision Operators:

eq or ==
ne or !=
lt or <
le or <=
gt or >
ge or >=

iv).Ternary Operator:

Expr1? Expr2: Expr3;

v).Type Operator[T]:

It able to represent a particular class type or interface type in SpEL.

Syntax:

T(Class_Name)

vi).Safe Navigation Operator:**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In general, in JAVA applications, if we access any instance variable or method on a reference variable contains null value then JVM will rise an exception like `java.lang.NullPointerException`. In SpEL , to avoid NullPointerExceptions we are able to use "Safe Navigation" operator.

Syntax:`var_Name?.methid_or_Var_Name()`**Example:**`package com.durgasoft.beans;`

```
public class User {  
    private String uname;
```

```
    public String getUsername() {  
        return uname;  
    }
```

```
    public void setUsername(String uname) {  
        this.uname = uname;  
    }
```

Test.java`package com.durgasoft.test;`

```
import org.springframework.expression.Expression;  
import org.springframework.expression.ExpressionParser;  
import org.springframework.expression.spel.standard.SpelExpressionParser;  
import org.springframework.expression.spel.support.StandardEvaluationContext;
```

CONTACT US:**Mobile:** +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import com.durgasoft.beans.User;

public class Test {

    public static void main(String[] args) throws Exception {
        ExpressionParser parser = new SpelExpressionParser();
        Expression expr = parser.parseExpression("10 + 20");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("10 * 20");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("true and true");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("true && false");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("true or true");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("true || false");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("10 ne 20");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("10 >= 20");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("10 lt 5");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("10 ge 5");
        System.out.println(expr.getValue());
    }
}
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
expr = parser.parseExpression("10 eq 10? 'condition is true': 'condition is  
false'");  
System.out.println(expr.getValue());  
  
expr = parser.parseExpression("T(Thread).MIN_PRIORITY");  
System.out.println(expr.getValue());  
  
expr = parser.parseExpression("T(Integer).toString(10)");  
System.out.println(expr.getValue());  
  
User u = new User();  
StandardEvaluationContext context = new StandardEvaluationContext(u);  
expr = parser.parseExpression("uname?.toUpperCase()");  
System.out.println(expr.getValue(context));  
  
}  
}
```

3. Variables

In SpEL, if we want to access any variable which is already existed in StandardEvaluationContext then we have to use the following syntax.

#var_Name.

If we want to set variable to StandardEvaluationContext then we have to use the following method.

```
public void setVariable(String var_Name, Object val)
```

Example:

MyMath.java

```
package com.durgasoft.beans;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class MyMath {  
    private int num1;  
    private int num2;  
  
    setXXX() and getXXX()  
  
    public int add() {  
        return num1+num2;  
    }  
    public int sub() {  
        return num1-num2;  
    }  
    public int mul() {  
        return num1*num2;  
    }  
}
```

Test.java

```
package com.durgasoft.test;  
import org.springframework.expression.Expression;  
import org.springframework.expression.ExpressionParser;  
import org.springframework.expression.spel.standard.SpelExpressionParser;  
import org.springframework.expression.spel.support.StandardEvaluationContext;  
import com.durgasoft.beans.MyMath;  
public class Test {  
    public static void main(String[] args) throws Exception {  
        MyMath math = new MyMath();  
        StandardEvaluationContext context = new  
StandardEvaluationContext(math);  
        context.setVariable("number1", 10);  
        context.setVariable("number2", 5);  
        ExpressionParser parser = new SpelExpressionParser();  
        Expression expr1 = parser.parseExpression("num1 = #number1");  
        Expression expr2 = parser.parseExpression("num2 = #number2");  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        System.out.println(math.add());
        System.out.println(math.sub());
        System.out.println(math.mul());
    }
}
```

4. Method Invocations

In SpEL, we are able to declare methods and we are able to access those methods as per the requirement.

If we want to declare an user defined method and if we want to access user defined method through an expression then we have to use the following steps.

1. Create StandardEvaluationContext object.

2. Register Method with a name.

```
public void registerFunction(String meth_Name, Method m)
```

3. Prepare an Expression with method call and access it.

Example:

MyString.java

```
package com.durgasoft.beans;
public class MyString {
    public static void reverseString(String str) {
        StringBuffer sb = new StringBuffer(str);
        System.out.println(sb.reverse());
    }
}
```

Test.java

```
package com.durgasoft.test;
import java.lang.reflect.Method;
import org.springframework.expression.Expression;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.expression.ExpressionParser;
import org.springframework.expression.spel.standard.SpelExpressionParser;
import org.springframework.expression.spel.support.StandardEvaluationContext;
public class Test {
    public static void main(String[] args) throws Exception {
        StandardEvaluationContext context = new StandardEvaluationContext();
        Class cls = Class.forName("com.durgasoft.beans.MyString");
        Method method = cls.getDeclaredMethod("reverseString", new
Class[]{java.lang.String.class});
        context.registerFunction("reverse", method);
        context.setVariable("str", "Durga Software Solutions");

        ExpressionParser parser = new SpelExpressionParser();
        Expression expr = parser.parseExpression("#reverse(#str)");
        expr.getValue(context);

        // Accessing Predefined methods
        expr = parser.parseExpression("new java.util.Date().toString()");
        System.out.println(expr.getValue());

        expr = parser.parseExpression("Durga Software
Solutions'.toUpperCase()");
        System.out.println(expr.getValue());
    }
}
```

5. Collections

In SpEL, we are able to declare Collections and we are able to access the content from Collection objects by using the following Expression Syntax.

Collection_Ref_Var.?Condition_Expression

Example:

City_State.java

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
package com.durgasoft.beans;

public class City_State {
    private String city;
    private String state;

    public City_State(String city, String state) {
        this.city = city;
        this.state = state;
    }
    setXXX() and getXXX()
    public String toString() {
        return city+"="+state;
    }
}
```

City_State_Collection.java

```
package com.durgasoft.beans;

import java.util.ArrayList;

public class City_State_Collection {
    private ArrayList<City_State> city_State = new ArrayList<City_State>();
    public ArrayList<City_State> getCity_State(){
        city_State.add(new City_State("Hyd","Tel"));
        city_State.add(new City_State("VJA","AP"));
        city_State.add(new City_State("Warangal","Tel"));
        city_State.add(new City_State("Vizag","AP"));
        city_State.add(new City_State("KNGR","Tel"));
        city_State.add(new City_State("TRPTI","AP"));
        return city_State;
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
package com.durgasoft.test;

import java.util.ArrayList;

import org.springframework.expression.Expression;
import org.springframework.expression.ExpressionParser;
import org.springframework.expression.spel.standard.SpelExpressionParser;
import org.springframework.expression.spel.support.StandardEvaluationContext;

import com.durgasoft.beans.City_State;
import com.durgasoft.beans.City_State_Collection;

public class Test {

    public static void main(String[] args) throws Exception {
        City_State_Collection collection = new City_State_Collection();
        StandardEvaluationContext context = new
StandardEvaluationContext(collection);
        ExpressionParser parser = new SpelExpressionParser();
        Expression expr = parser.parseExpression("city_State.? [state == 'AP']");
        ArrayList<City_State> al = (ArrayList<City_State>)expr.getValue(context);
        System.out.println(al);

    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGA SOFT { SPRING JDBC MODULE }

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftware.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING JDBC MODULE INDEX

- | | |
|--|----------|
| 1. Spring-DAO..... | PAGE 238 |
| 2. Spring JDBC..... | PAGE 256 |
| 3. Spring JDBC Connection Pooling Mechanism..... | PAGE 301 |

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring-DAO

- >DAO [Data Access Object], it is a design pattern, it able to provide very good environment to separate Data Access logic from Business Processing logic.
- >In enterprise Applications, to prepare Data Access Layer we will use DAO Design pattern.

Advantages of DAOs in Enterprise Applications:

- 1)We are able to hide all data access implementation from Business/ Service Layer.
- 2)It is very simple to switch data access layer from one data access tech to another data access tech. without giving any effect to Service/Business Layer, that is from JDBC to Hibernate,....
- 3)DAOs are able to provide centralized Data Access in enterprise Application, it simplifies the maintenance of Enterprise Applications.
- 4)While preparing Service/Business layer Code Simplification is possible, that is, it is very simple to prepare Service/Business layer.
- 5)We are able to get Standardization to prepare Data Access layer with DAOs.

Drawbacks With DAOs:

- 1)It adds one more layer to enterprise Application, may get maintenance problems.
- 2)Along with DAOs, we have to implement some other Design patterns like Factory Classes, DTO[Data Transfer Objects],....in enterprise applications.

Guidelines to prepare DAOs in Enterprise Applications:

1. Prepare a separate DAO interface:

Prepare a separate DAO interface with the required DAO methods, which must represent CRUD operations.

EX:

```
public interface StudentDao{  
    public String addStudent(Student std) throws DAOException;  
    public String updateStudent(Student std) throws DAOException;  
    public String deleteStudent(String sid) throws DAOException;  
    public List<Student> findAllStudents() throws DAOException;  
    public Student findStudentByID(String sid) throws DAOException;  
    public Student findStudentByName(String sname) throws DAOException;  
    ----  
    ----
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

2. Provide an implementation to DAO interface:

```
public class StudentDAOImpl implements StudentDao{  
---implementation for all StudentDao interface methods-----  
---> We can provide our methods inorder to improve code reusability along with DAO interface  
methods---  
}
```

3. Prepare DTOs as per the requirement:

```
public class Student{  
private String sid;  
private String sname;  
private String saddr;  
----  
----  
setXXX() and getXXX()  
----  
----  
}
```

4. Create Factory Methods/ Factory Classes to generate DAOs:

```
public class StudentDaoFactory{  
private static Student Dao dao;  
static{  
dao = new StudentDAOImpl();  
}  
public static StudentDao getStudentDao(){  
return dao;  
}  
}
```

In Service layer
StudentDao dao = StudentDao.getStudentDao();

5) We must not cache DAO references, because, Factory classes/ Factory methods are providing single instances of DAO to the service layer, if DAO is required in multiple modules then it is required to create more than one DAO reference.

6) In case of DAOs, It is suggestible to interact with Databases by using Connection Pooling mechanisms, not by using DriverManager approach.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 7) DAO is not threadsafe, we must not use DAOs in multi threaded environment.
- 8) In DAOs we can access close() method inorder to close the resources like connections,... , so here, before calling close() method we must ensure that whether the resources are going to be released or not with our close() method call.
- 9) We have make sure that all the objects which are used by DAOs are following Java bean conventions or not.

Application on Servlets and JSPs with DAO:

Resources:

1. htmls:
 - a)addform.html
 - b)searchform.html
 - c)deleteform.html
 - d)existed.html
 - e)notexisted.html
 - f)success.html
 - g)failure.html
 - h)layout.html
 - i)header.html
 - j)menu.html
 - k)welcome.html
 - l)footer.html
- 2.jspss:
 - a)display.jsp
- 3.Servlets:
 - a)ControllerServlet
- 4.Services:
 - a)StudentService
- 5.DAOs:
 - a)StudentDao
- 6.DTOs:
 - a)StudentTo
- 7)Factories:
 - a)ConnectionFactory
 - b)StudentServiceFactory
 - c)StudentDaoFactory
- 8)JARS:

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

a)ojdbc6.jar

Design Patterns:

- a)DAO
- b)MVC
- c)DTO
- d)Factory

Example:**layout.html**

```
<frameset rows="20%,65%,15%">
    <frame src="header.html"/>
    <frameset cols="20%,80%">
        <frame src="menu.html"/>
        <frame src="welcome.html" name="body"/>
    </frameset>
    <frame src="footer.html"/>
</frameset>
```

header.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor="maroon">
<center>
<font color="white" size="7">
<b>
DURGA SOFTWARE SOLUTIONS
</b>
</font>
</center>
</body>
</html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

footer.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">  
<html>  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">  
<title>Insert title here</title>  
</head>  
<body bgcolor="blue">  
<center>  
<font color="white" size="5">  
<b>  
Durga Software Solutions, 202, Mitrivanam, Ameerpet, Hyd-38  
</b>  
</font>  
</center>  
</body>  
</html>
```

menu.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">  
<html>  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">  
<title>Insert title here</title>  
</head>  
<body bgcolor="lightyellow">  
<center>  
<h3>  
<br><br>  
<a href=".//addform.html" target="body">Add Student</a><br><br>  
<a href=".//searchform.html" target="body">Search Student</a><br><br>  
<a href=".//deleteform.html" target="body">Delete Student</a>  
</h3>  
</center>  
</body>  
</html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

welcome.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor="lightblue">
<center>
<br><br><br>
<font color="red" size="6">
<b>
<marquee>
Welcome To Durga Software Solutions
</marquee>
</b>
</font>
</center>
</body>
</html>
```

addform.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor="lightblue">
<form method="POST" action=".controller">
<center>
<br><br><br>
<table>
<tr>
<td>Student Id</td><td><input type="text" name="sid"/></td>
</tr>
<tr>
<td>Student Name</td><td><input type="text" name="sname"/></td>
</tr>
</table>
</center>
</form>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<tr>
    <td>Student Address</td><td><input type="text" name="saddr"/></td>
</tr>
<tr>
    <td><input type="submit" value="ADD" name="button"/>
</tr>
</table>
</center>
</form>
</body>
</html>
```

searchform.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor="lightblue">
<form method="POST" action=".controller">
<br><br><br>
<center>
<table>
<tr>
    <td>Student Id</td><td><input type="text" name="sid"/></td>
</tr>
<tr>
    <td><input type="submit" value="SEARCH" name="button"/></td>
</tr>
</table>
</center>
</form>
</body>
</html>
```

deleteform.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor="lightblue">
<form method="POST" action=".controller">
<br><br><br>
<center>
<table>
<tr>
    <td>Student Id</td><td><input type="text" name="sid"/></td>
</tr>
<tr>
    <td><input type="submit" value="DELETE" name="button"/></td>
</tr>
</table>
</center>
</form>
</body>
</html>
```

display.jsp

```
<%@page import="com.durgasoft.to.StudentTo"%>
<%!
StudentTo sto;
%>
<%
sto = (StudentTo)request.getAttribute("sto");
%>
<html>
<body bgcolor="lightblue">
<center>
<br><br><br>
<table border = "1" bgcolor="white">
<tr>
    <td>Student Id</td><td><%= sto.getSid() %></td>
</tr>
<tr>
    <td>Student Name</td><td><%= sto.getSname() %></td>
</tr>
<tr>
    <td>Student Address</td><td><%= sto.getSaddr() %></td>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftwareonlinetraining@gmail.com

WEBSITE: www.durgasoftwareonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</tr>  
  
</table>  
</center>  
</body>  
</html>
```

existed.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">  
<html>  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">  
<title>Insert title here</title>  
</head>  
<body bgcolor="lightblue">  
<center>  
<br><br><br>  
<font color="red" size="6">  
<b>  
Student Existed Already  
</b>  
</font>  
</center>  
</body>  
</html>
```

notexisted.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">  
<html>  
<head>  
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">  
<title>Insert title here</title>  
</head>  
<body bgcolor="lightblue">  
<center>  
<br><br><br>  
<font color="red" size="6">  
<b>  
Student Not Existed  
</b>  
</font>  
</center>  
</body>  
</html>
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</b>
</font>
</center>
</body>
</html>
```

success.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor="lightblue">
<center>
<br><br><br>
<font color="red" size="6">
<b>
Success
</b>
</font>
</center>
</body>
</html>
```

failure.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body bgcolor="lightblue">
<center>
<br><br><br>
<font color="red" size="6">
<b>
Failure
</b>
</font>
</center>
</body>
</html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</font>
</center>
</body>
</html>
```

ControllerServlet.java

```
package com.durgasoft.controller;
import java.io.IOException;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.durgasoft.factory.StudentServiceFactory;
import com.durgasoft.services.StudentService;
import com.durgasoft.to.StudentTo;
public class ControllerServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        String button_Label = request.getParameter("button");
        String status = "";
        RequestDispatcher rd = null;
        if(button_Label.equals("ADD")){
            StudentService service = StudentServiceFactory.getStudentService();
            StudentTo sto = new StudentTo();
            sto.setSid(request.getParameter("sid"));
            sto.setSname(request.getParameter("sname"));
            sto.setSaddr(request.getParameter("saddr"));
            status = service.addStudent(sto);
            if(status.equals("success")){
                rd = request.getRequestDispatcher("./success.html");
                rd.forward(request, response);
            }
            if(status.equals("failure")){
                rd = request.getRequestDispatcher("./failure.html");
                rd.forward(request, response);
            }
            if(status.equals("existed")){
                rd = request.getRequestDispatcher("./exist.html");
                rd.forward(request, response);
            }
        }
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        rd = request.getRequestDispatcher("./existed.html");
        rd.forward(request, response);
    }
}

if(button_Label.equals("SEARCH")){
    String sid = request.getParameter("sid");
    StudentService service = StudentServiceFactory.getStudentService();
    StudentTo sto = service.searchStudent(sid);
    RequestDispatcher dispatcher = null;
    if( sto == null ) {
        dispatcher = request.getRequestDispatcher("notexisted.html");
        dispatcher.forward(request, response);
    }else {
        request.setAttribute("sto", sto);
        dispatcher = request.getRequestDispatcher("display.jsp");
        dispatcher.forward(request, response);
    }
}

if(button_Label.equals("DELETE")){
    String sid = request.getParameter("sid");
    StudentService service = StudentServiceFactory.getStudentService();
    status = service.deleteStudent(sid);
    RequestDispatcher dispatcher = null;
    if( status.equals("success") ) {
        dispatcher = request.getRequestDispatcher("success.html");
        dispatcher.forward(request, response);
    }
    if( status.equals("failure") ) {
        dispatcher = request.getRequestDispatcher("failure.html");
        dispatcher.forward(request, response);
    }
    if( status.equals("notexisted") ) {
        dispatcher = request.getRequestDispatcher("notexisted.html");
        dispatcher.forward(request, response);
    }
}
```





BY NAGOOR BABU

StudentService.java

```
package com.durgasoft.services;

import com.durgasoft.to.StudentTo;

public interface StudentService {
    public String addStudent(StudentTo sto);
    public StudentTo searchStudent(String sid);
    public String deleteStudent(String sid);
}
```

StudentServiceImpl.java

```
package com.durgasoft.services;

import com.durgasoft.dao.StudentDao;
import com.durgasoft.factory.StudentDaoFactory;
import com.durgasoft.to.StudentTo;

public class StudentServiceImpl implements StudentService {
    String status="";
    @Override
    public String addStudent(StudentTo sto) {
        StudentDao dao = StudentDaoFactory.getStudentDao();
        status = dao.add(sto);
        return status;
    }

    @Override
    public StudentTo searchStudent(String sid) {
        StudentTo sto = null;
        StudentDao dao = StudentDaoFactory.getStudentDao();
        sto = dao.search(sid);
        return sto;
    }

    @Override
    public String deleteStudent(String sid) {
        StudentDao dao = StudentDaoFactory.getStudentDao();
        status = dao.delete(sid);
        return status;
    }
}
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

}

StudentDao.java

```
package com.durgasoft.dao;  
  
import com.durgasoft.to.StudentTo;  
  
public interface StudentDao {  
    public String add(StudentTo sto);  
    public StudentTo search(String sid);  
    public String delete(String sid);  
}
```

StudentDaolmpl.java

```
package com.durgasoft.dao;  
  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
  
import com.durgasoft.factory.ConnectionFactory;  
import com.durgasoft.to.StudentTo;  
  
public class StudentDaolmpl implements StudentDao{  
    String status = "";  
    @Override  
    public String add(StudentTo sto) {  
        try {  
            Connection con = ConnectionFactory.getConnection();  
            PreparedStatement pst = con.prepareStatement("select * from student where  
sid = ?");  
            pst.setString(1, sto.getSid());  
            ResultSet rs = pst.executeQuery();  
            boolean b = rs.next();  
            if( b == true ) {  
                status = "existed";  
            }else {  
                pst = con.prepareStatement("insert into student values(?, ?, ?)");  
                pst.setString(1, sto.getSid());  
                pst.setString(2, sto.getsName());  
                pst.setString(3, sto.getsAddr());  
            }  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

        pst.executeUpdate();
        status = "success";
    }
} catch (Exception e) {
    status = "failure";
    e.printStackTrace();
}
return status;
}

@Override
public StudentTo search(String sid) {
    StudentTo sto = null;
    try {
        Connection con = ConnectionFactory.getConnection();
        PreparedStatement pst = con.prepareStatement("select * from student where
sid = ?");
        pst.setString(1, sid);
        ResultSet rs = pst.executeQuery();
        boolean b = rs.next();
        if(b == true) {
            sto = new StudentTo();
            sto.setSid(rs.getString("SID"));
            sto.setSname(rs.getString("SNAME"));
            sto.setSaddr(rs.getString("SADDR"));
        }else {
            sto = null;
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return sto;
}

@Override
public String delete(String sid) {
    try {
        Connection con = ConnectionFactory.getConnection();
        PreparedStatement pst = con.prepareStatement("select * from student where
sid = ?");
        pst.setString(1, sid);
        ResultSet rs = pst.executeQuery();
        boolean b = rs.next();
        if(b == true) {
            pst = con.prepareStatement("delete from student where sid = ?");
        }
    }
}

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        pst.setString(1, sid);
        pst.executeUpdate();
        status = "success";
    }else {
        status = "notexisted";
    }
} catch (Exception e) {
    status = "failure";
    e.printStackTrace();
}
return status;
}
```

StudentTo.java

```
package com.durgasoft.to;

public class StudentTo {
    private String sid;
    private String sname;
    private String saddr;

    public String getSid() {
        return sid;
    }
    public void setSid(String sid) {
        this.sid = sid;
    }
    public String getSname() {
        return sname;
    }
    public void setSname(String sname) {
        this.sname = sname;
    }
    public String getSaddr() {
        return saddr;
    }
    public void setSaddr(String saddr) {
        this.saddr = saddr;
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

StudentServiceFactory.java

```
package com.durgasoft.factory;

import com.durgasoft.services.StudentService;
import com.durgasoft.services.StudentServiceImpl;

public class StudentServiceFactory {
    private static StudentService service;
    static{
        service = new StudentServiceImpl();
    }
    public static StudentService getStudentService(){
        return service;
    }
}
```

StudentDaoFactory.java

```
package com.durgasoft.factory;

import com.durgasoft.dao.StudentDao;
import com.durgasoft.dao.StudentDaoImpl;

public class StudentDaoFactory {
    private static StudentDao dao;
    static {
        dao = new StudentDaoImpl();
    }
    public static StudentDao getStudentDao() {
        return dao;
    }
}
```

ConnectionFactory.java

```
package com.durgasoft.factory;

import java.sql.Connection;
import java.sql.DriverManager;

public class ConnectionFactory {
    private static Connection con;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

static {
    try {
        Class.forName("oracle.jdbc.OracleDriver");
        con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe",
"system", "durga");
    } catch (Exception e) {
        e.printStackTrace();
    }
}

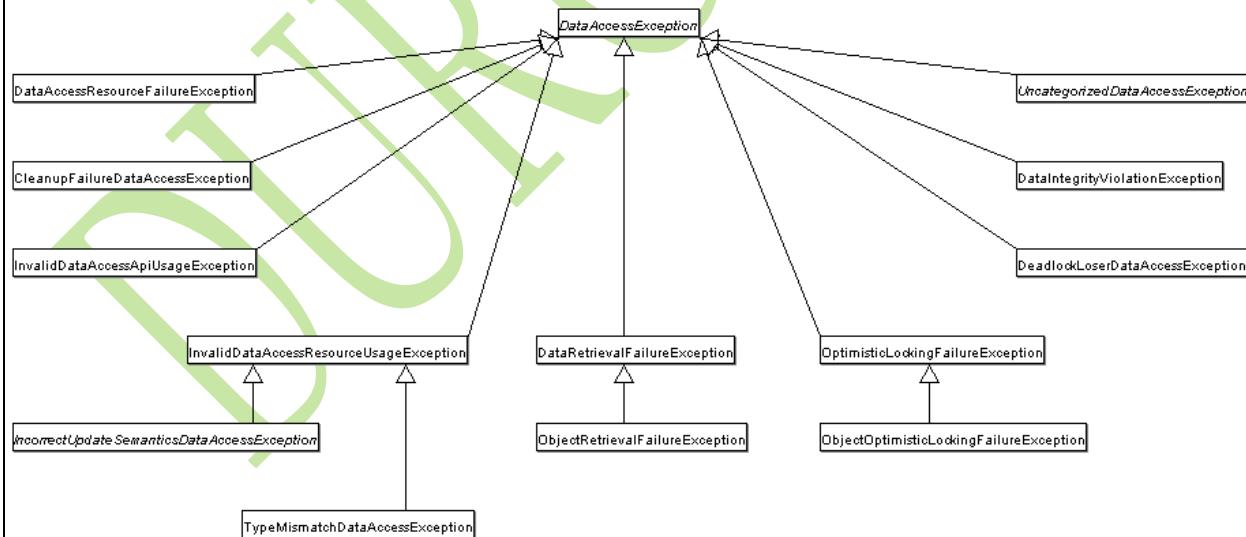
public static Connection getConnection() {
    return con;
}
}

```

To provide support for DAOs kind of implementations in Spring Applications, Spring has provided a separate module called as “Spring DAO”.

Spring DAO modules has provided a set of predefined classes and interfaces in order to provide DAO support in the form of “org.springframework.dao” package.

Spring provides a convenient translation from technology-specific exceptions like `SQLException` , `HibernateException`,..... to its own exception class hierarchy with the `DataAccessException` as the root exception.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring JDBC

In Enterprise Applications, to prepare Data Access Layer or DAO layer Spring has provided Modules in the form of JDBC and ORM . IN ORM we may use no of ORM implementation tools like Hibernate, JPA, Ibatis,....

Q)In Enterprise Applications, to prepare Data Access Layer we have already Plain JDBC tech. then what is the requirement to go for Spring JDBC Module?

Ans:

1.To prepare Data Access Layer in enterprise applications, if we use JDBC then we must take explicit responsibility to prepare the steps load and register the driver, Establishing Connection, creating Statement, executing SQL Queries and closing the resources like ResultSet, Statement and Connection.

If we use Spring JDBC module to prepare Data Access Layer, we must take explicit responsibility to write and execute SQL Queries only, not to take any responsibility to load and register driver, connection establishment, creating Statement and closing the resources.

2.In case of Plain JDBC, almost all the exceptions are checked exceptions, we have to handle them explicitly by providing some java code.

In case of Spring JDBC module, all the internal checked exceptions are converted into Unchecked Exceptions which are defined by Spring DAO module , it is very simple to handle these unchecked Exceptions.

3.In Plain JDBC, limited support is available for Transactions.

In Spring JDBC Module, very good support is available for transactions, we may use Transaction module also to provide transactions.

4.In Plain JDBC, to hold the results we are able to use only ResultSet object, which is not implementing java.io.Serializable interface, which is not transferable in network.

In Spring JDBC, we are able to get results of SQL Queries in our required form like in the form of RowSet, Collections, which are implementing java.io.Serializable interface and which are transferable in Network.

5.In plain JDBC, we are able to get Connections either by using DriverManager or by using Datasource.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In Spring JDBC, we are able to get Connection internally by using Datasource only, that is through Connection Pooling only.

6.In plain JDBC, to map records to Bean objects in the form of Collection Object we have to write java code explicitly, no predefined support is provided by JDBC tech.

In case of Spring JDBC, to map Database records to Bean objects in the form of Collection Spring JDBC has provided predefined support in the form of "RowMapper".

7.In Plain JDBC, no callback interfaces support is available to create and execute the sql queries in PreparedStatement style.

In Spring JDBC, callback interfaces support is available to create and execute sql queries in PreparedStatement style.

To prepare Data Access Layer in enterprise applications, Spring JDBC module has provided the complete predefined library in the from of the following classes and interfaces in "org.springframework.jdbc" and its sub packages.

JdbcTemplate
NamedParameterJdbcTemplate
SimpleJdbcTemplate
SimpleJdbcInsert and SimpleJdbcCall
SQL Mapping through SQLUpdate and SQLInsert

In Spring Applications, if we want to JdbcTemplate[Jdbc Module] then we have to use the following steps.

- 1)Create DAO interface with the required methods.
- 2)Create DAO implementation class with implementation for DAO interface methods.
- 3)In Configuration File provide configuration for DataSource class , JdbcTemplate class and DAO implementation class.
- 4)Prepare Test Application to access Dao methods.

IN Spring configuration file we have to configure DataSource with the following properties.

driverClassName
url
username
password

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In Spring applications, to configure DataSource Spring has provided a separate a predefined Datasource class in the form of "org.springframework.jdbc.datasource.DriverManagerDataSource", it is not suggestible for production mode, it is suggestible for testing mode of our applications , In spring applications, it is always suggestible to use third party Connection Pooling mechanisms like dbcp, C3P0, Proxool,..

JdbcTemplate class is providing basic environment to interact with Database like Loading Driver class, Getting Connection between Java application and DB, Creating Statement , PreparedStatement and CallableStatement and closing the connection with the help of the provided Datasource and JdbcTemplate class has provided the following methods to execute SQL Queries.

1)For Non Select sql queries and DML SQL queries

```
public int update(String query)
```

2)For DDL Sql Queries

```
public void execute(String query)
```

3)For Select sql queries

```
public int queryForInt(String query)
public int queryForLong(String query)
public String queryForString(String query)
public Object queryForObject(String query)
public List query(String query)
public List queryForList(String query)
public Map queryForMap(String query)
public RowSet queryForRowSet(String query)
```

While performing retrieval operations to convert data from ResultSet object[records] to Bean objects Spring Framework has provided a predefined interface in the form of "org.springframework.jdbc.core.RowMapper" which contains the following method .

```
public Object mapRow(ResultSet rs, int rowCount)
```

Example

StudentDao.java

```
package com.durgasoft.dao;
import org.springframework.jdbc.core.JdbcTemplate;
import com.durgasoft.beans.Student;
public interface StudentDao {
    public void setJdbcTemplate(JdbcTemplate jdbcTemplate);
    public String add(Student std);
    public Student search(String sid);
    public String update(Student std);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public String delete(String sid);  
}
```

StudentDaoImpl.java

```
package com.durgasoft.dao;  
  
import org.springframework.jdbc.core.JdbcTemplate;  
  
import com.durgasoft.beans.Student;  
  
public class StudentDaoImpl implements StudentDao {  
    private JdbcTemplate jdbcTemplate;  
    String status = "";  
  
    @Override  
    public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
        this.jdbcTemplate = jdbcTemplate ;  
    }  
    @Override  
    public String add(Student std) {  
        try {  
            jdbcTemplate.update("insert into student  
values ('"+std.getSid()+"','"+std.getSname()+"','"+std.getSaddr()+"')");  
            status = "success";  
        }catch(Exception e) {  
            status = "failure";  
            e.printStackTrace();  
        }  
        return status;  
    }  
    @Override  
    public Student search(String sid) {  
        Student std = null;  
        try {
```

CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28 **US NUM:** 4433326786**Mail ID:** durgasoftonlinetraining@gmail.com**WEBSITE:** www.durgasoftonline.com**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
        std = jdbcTemplate.queryForObject("select * from student where  
        sid='"+sid+"'", new StudentMapper());  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
    return std;  
}  
  
@Override  
public String update(Student std) {  
    try {  
        jdbcTemplate.update("update student set  
        sname='"+std.getSname()+"',saddr='"+std.getSaddr()+"' where sid='"+std.getId()+"');  
        status="success";  
        jdbcTemplate.qu  
    }catch(Exception e) {  
        status="failure";  
        e.printStackTrace();  
    }  
    return status;  
}  
  
@Override  
public String delete(String sid) {  
    try {  
        jdbcTemplate.update("delete from student where sid='"+sid+"');  
        status="success";  
    } catch (Exception e) {  
        status="failure";  
        e.printStackTrace();  
    }  
    return status;  
}  
  
Student.java  
  
package com.durgasoft.beans;  
  
public class Student {  
    private String sid;  
    private String sname;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private String saddr;  
  
    setXXX() and getXXX()  
}
```

StudentMapper.java

```
package com.durgasoft.dao;  
  
import java.sql.ResultSet;  
import java.sql.SQLException;  
  
import org.springframework.jdbc.core.RowMapper;  
  
import com.durgasoft.beans.Student;  
  
public class StudentMapper implements RowMapper<Student> {  
    @Override  
    public Student mapRow(ResultSet rs, int row_No) throws SQLException {  
  
        Student std = new Student();  
        std.setSid(rs.getString("SID"));  
        std.setSname(rs.getString("SNAME"));  
        std.setSaddr(rs.getString("SADDR"));  
        return std;  
    }  
}
```

Test.java

```
package com.durgasoft.test;  
  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
import com.durgasoft.beans.Student;  
import com.durgasoft.dao.StudentDao;  
  
public class Test {  
  
    public static void main(String[] args) throws Exception {
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28 **US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
StudentDao dao = (StudentDao) context.getBean("studentDao");
-----Inserting Records-----
Student std = new Student();
std.setSid("S-111");
std.setSname("Durga");
std.setSaddr("Hyd");
String status = dao.add(std);
System.out.println("Student Insertion :" +status);

std.setSid("S-222");
std.setSname("Anil");
std.setSaddr("Hyd");
status = dao.add(std);
System.out.println("Student Insertion :" +status);

std.setSid("S-333");
std.setSname("Sekhar");
std.setSaddr("Hyd");
status = dao.add(std);
System.out.println("Student Insertion :" +status);
System.out.println();

-----Retrieving Record-----
Student std1 = dao.search("S-111");
if(std1 == null) {
    System.out.println("Student Search Status :NotExisted");
}else {
    System.out.println("Student Details");
    System.out.println("-----");
    System.out.println("Student Id    :" +std1.getSid());
    System.out.println("Student Name   :" +std1.getSname());
    System.out.println("Student Address :" +std1.getSaddr());
}
System.out.println();

-----Updating a Record-----
std.setSid("S-111");
std.setSname("XXX");
std.setSaddr("YYY");
status = dao.update(std);
System.out.println("Student Updation :" +status);
System.out.println();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
    //---Deleting a record----  
    status = dao.delete("S-111");  
    System.out.println("Student Deletion :" +status);  
}  
}
```

In Spring JDBC Applications, we will use positional parameters[?] also in sql queries which we are providing along with JdbcTemplate class provided query execution methods.

If we provide positional parameters in sql queries then JdbcTemplate class will use "PreparedStatement" internally to execute sql query instead of Statement.

To provide values to the Positional parameters in SQL Queries we have to use Object[] with values as parameter to all JdbcTemplate class provided query execution methods.

```
public int update(String query, Object[] param_Values)  
public int queryForInt(String query, Object[] param_Values)  
public long queryForLong(String query, Object[] param_Values)  
public Object queryForObject(String query, Object[] param_Values, RowMapper rm)  
-----  
-----  
-----
```

Ex:

```
String query = "insert into student values(?, ?, ?);  
int rowCount = jdbcTemplate.update(query, new Object[]{"S-111", "AAA", "Hyd"});
```

Example:

StudentDao.java

```
package com.durgasoft.dao;  
  
import org.springframework.jdbc.core.JdbcTemplate;  
  
import com.durgasoft.beans.Student;  
  
public interface StudentDao {  
    public void setJdbcTemplate(JdbcTemplate jdbcTemplate);  
    public String add(Student std);  
    public Student search(String sid);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
    public String update(Student std);
    public String delete(String sid);

}
```

StudentDaoImpl.java

```
package com.durgasoft.dao;

import org.springframework.jdbc.core.JdbcTemplate;
import com.durgasoft.beans.Student;

public class StudentDaoImpl implements StudentDao {
    private JdbcTemplate jdbcTemplate;
    String status = "";

    @Override
    public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
        this.jdbcTemplate = jdbcTemplate ;
    }

    @Override
    public String add(Student std) {
        try {
            String query = "insert into student values(?, ?, ?)";
            jdbcTemplate.update(query, new Object[] {std.getId(), std.getName(),
                std.getAddr()});
            status = "success";
        }catch(Exception e) {
            status = "failure";
            e.printStackTrace();
        }
        return status;
    }

    @Override
    public Student search(String sid) {
```

CONTACT US:**Mobile: +91- 8885 25 26 27**

+91- 7207 21 24 27/28

**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```

Student std = null;
try {
    std = jdbcTemplate.queryForObject("select * from student where sid=?", new
Object[] {sid}, new StudentMapper());
} catch (Exception e) {
    e.printStackTrace();
}
return std;
}

@Override
public String update(Student std) {
try {
    jdbcTemplate.update("update student set sname=?, saddr=? where
sid=?",
new Object[] { std.getSname(), std.getSaddr(), std.getId()});
status="success";
} catch (Exception e) {
    status="failure";
    e.printStackTrace();
}
return status;
}

@Override
public String delete(String sid) {
try {
    jdbcTemplate.update("delete from student where sid=?", new Object[] {sid});
status="success";
} catch (Exception e) {
    status="failure";
    e.printStackTrace();
}
return status;
}

```

applicationContext.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd">
```

```
<bean id="studentDao" class="com.durgasoft.dao.StudentDaoImpl">
    <property name="jdbcTemplate" ref="jdbcTemplate"/>
</bean>
<bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
    <property name="dataSource" ref="dataSource"/>
</bean>

<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource" >
    <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
    <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
    <property name="username" value="system"/>
    <property name="password" value="durga"/>
</bean>
```

```
</beans>
```

StudentMapper.java

```
package com.durgasoft.dao;
import java.sql.ResultSet;
import java.sql.SQLException;
import org.springframework.jdbc.core.RowMapper;
import com.durgasoft.beans.Student;
public class StudentMapper implements RowMapper<Student> {
@Override
public Student mapRow(ResultSet rs, int row_No) throws SQLException {
    Student std = new Student();
    std.setSid(rs.getString("SID"));
    std.setSname(rs.getString("SNAME"));
    std.setSaddr(rs.getString("SADDR"));
}
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        return std;  
  
    }  
}
```

Test.java

```
package com.durgasoft.test;  
  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
import com.durgasoft.beans.Student;  
import com.durgasoft.dao.StudentDao;  
  
public class Test {  
  
    public static void main(String[] args) throws Exception {  
        ApplicationContext context = new  
ClassPathXmlApplicationContext("applicationContext.xml");  
        StudentDao dao = (StudentDao) context.getBean("studentDao");  
        //----Inserting Records-----  
        Student std = new Student();  
        std.setSid("S-111");  
        std.setSname("Durga");  
        std.setSaddr("Hyd");  
        String status = dao.add(std);  
        System.out.println("Student Insertion :" + status);  
  
        std.setSid("S-222");  
        std.setSname("Anil");  
        std.setSaddr("Hyd");  
        status = dao.add(std);  
        System.out.println("Student Insertion :" + status);  
  
        std.setSid("S-333");  
        std.setSname("Sekhar");  
        std.setSaddr("Hyd");  
        status = dao.add(std);  
        System.out.println("Student Insertion :" + status);  
        System.out.println();  
  
        //----Retriving Record-----  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
Student std1 = dao.search("S-111");
if(std1 == null) {
    System.out.println("Student Search Status :NotExisted");
}else {
    System.out.println("Student Details");
    System.out.println("-----");
    System.out.println("Student Id    :" +std1.getId());
    System.out.println("Student Name   :" +std1.getName());
    System.out.println("Student Address :" +std1.getAddress());
}
System.out.println();

//----Updating a Record-----
std.setId("S-111");
std.setName("XXX");
std.setAddress("YYY");
status = dao.update(std);
System.out.println("Student Updation :" +status);
System.out.println();

//----Deleting a record-----
status = dao.delete("S-111");
System.out.println("Student Deletion :" +status);
}
```

NamedParameterJdbcTemplate

NamedParameterJdbcTemplate class is same as JdbcTemplate class , but, NamedParameterJdbcTemplate class is able to define and run sql queries with Named Parameters instead of positional parameters.

EX:

```
String query = "insert into student values(:sid, :name, :address);"
```

Where :sid, :name, :address are named parameters for which we have to provide values.

In case of NamedParameterJdbcTemplate , we are able to provide values to the named parameters in the following two approaches.

- 1.By Using Map directly.
- 2.By using SqlParameterSource interface.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1.By Using Map directly.

```
String query = "insert into student values(:sid, :sname, :saddr)";  
Map map = new HashMap();  
map.put("sid", "S-111");  
map.put("sname", "AAA");  
map.put("saddr", "Hyd");  
namedParameterJdbcTemplate.update(query, map);
```

2.By using SqlParameterSource interface.

To provide values to the Named parameters Spring has provided the following two implementation classes for SqlParameterSource interface.

- a)MapSqlParameterSource
- b)BeanPropertySqlParameterSource

To provide values to the named parameters if we want to use MapSqlParameterSource then first we have to create object for MapSqlParameterSource and we have to use the following method to add values to the named parameters.

```
public MapSqlParameterSource.addValue(String name, Object val)
```

EX:

```
String query = "insert into student values(:sid, :sname, :saddr)";  
SqlParameterSource param_Source = new MapSqlParameterSource("sid", "S-111");  
param_Source = param_Source.addValue("sname", "AAA");  
param_Source = param_Source.addValue("saddr", "Hyd");  
namedParameterJdbcTemplate.update(query, param_Source);
```

To provide values to the named parameters if we want to use BeanPropertySqlParameterSource then first we have to create bean object with data then we have to create Object for BeanPropertySqlParameterSource with the generated Bean reference then provide BeanPropertySqlParameterSource object to query methods.

EX:

```
String query = "insert into student values(:sid, :sname, :saddr)";  
Student std = new Student();  
std.setSid("S-111");  
std.setSname("AAA");  
std.setSaddr("Hyd");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
SqlParameterSource param_Source = new BeanPropertySqlParameterSource(std );
namedParameterJdbcTemplate.update(query, param_Source);
```

Note: JdbcTemplate is allowing DataSource object injection through setter method, but, NamedParameterJdbcTemplate class is allowing DataSource object injection through Constructor Dependency Injection.

Example

CustomerDao.java

```
package com.durgasoft.dao;

import com.durgasoft.beans.Customer;

public interface CustomerDao {

    public String add(Customer c);
    public Customer search(String cid);
    public String update(Customer c);
    public String delete(String cid);
}
```

CustomerDaoImpl.java

```
package com.durgasoft.dao;

import java.util.HashMap;
import java.util.Map;

import org.springframework.jdbc.core.namedparam.BeanPropertySqlParameterSource;
import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import org.springframework.jdbc.core.namedparam.SqlParameterSource;
import org.springframework.jdbc.datasource.DriverManagerDataSource;

import com.durgasoft.beans.Customer;

public class CustomerDaolmpl implements CustomerDao {
    String status = "";
    private NamedParameterJdbcTemplate namedParameterJdbcTemplate;
    public void setNamedParameterJdbcTemplate(NamedParameterJdbcTemplate
namedParameterJdbcTemplate) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

        this.namedParameterJdbcTemplate = namedParameterJdbcTemplate;
    }

    @Override
    public String add(Customer c) {
        String query = "insert into customer values(:cid, :cname, :caddr)";
        Map<String, Object> map = new HashMap<String, Object>();
        map.put("cid", c.getCid());
        map.put("cname", c.getCname());
        map.put("caddr", c.getAddr());
        namedParameterJdbcTemplate.update(query, map);
        return "SUCCESS";
    }

    @Override
    public Customer search(String cid) {
        String query = "select * from customer where cid=:cid";

        Map<String, Object> map = new HashMap<String, Object>();
        map.put("cid", cid);
        //SqlParameterSource paramSource = new MapSqlParameterSource("cid", cid);

        Customer c = namedParameterJdbcTemplate.queryForObject(query, map, new
CustomerMapper());
        return c;
    }

    @Override
    public String update(Customer c) {
        String query = "update customer set CNAME=:cname, CADDR=:caddr where
CID=:cid";
        SqlParameterSource paramSource = new BeanPropertySqlParameterSource(c);
        namedParameterJdbcTemplate.update(query, paramSource);

        return "SUCCESS";
    }

    @Override
    public String delete(String cid) {
        String query = "delete from customer where cid=:cid";
        SqlParameterSource paramSource = new MapSqlParameterSource("cid", cid);
        namedParameterJdbcTemplate.update(query, paramSource);
        return "SUCCESS";
    }
}

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftware.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

Customer.java

```
package com.durgasoft.beans;

public class Customer {
    private String cid;
    private String cname;
    private String caddr;

    public String getCid() {
        return cid;
    }
    public void setCid(String cid) {
        this.cid = cid;
    }
    public String getCname() {
        return cname;
    }
    public void setCname(String cname) {
        this.cname = cname;
    }
    public String getCaddr() {
        return caddr;
    }
    public void setCaddr(String caddr) {
        this.caddr = caddr;
    }
}
```

CustomerMapper.java

```
package com.durgasoft.dao;

import java.sql.ResultSet;
import java.sql.SQLException;

import org.springframework.jdbc.core.RowMapper;

import com.durgasoft.beans.Customer;

public class CustomerMapper implements RowMapper<Customer> {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
@Override  
public Customer mapRow(ResultSet rs, int row_No) throws SQLException {  
  
    Customer c = new Customer();  
    c.setCid(rs.getString("CID"));  
    c.setCname(rs.getString("CNAME"));  
    c.setCaddr(rs.getString("CADDR"));  
    return c;  
  
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
       xmlns:context="http://www.springframework.org/schema/context"  
       xsi:schemaLocation="  
           http://www.springframework.org/schema/beans  
           http://www.springframework.org/schema/beans/spring-beans.xsd  
           http://www.springframework.org/schema/context  
           http://www.springframework.org/schema/context/spring-context.xsd">  
  
<bean id="customerDao" class="com.durgasoft.dao.CustomerDaoImpl">  
    <property name="namedParameterJdbcTemplate" ref="namedParameterJdbcTemplate"/>  
</bean>  
    <bean id="namedParameterJdbcTemplate"  
          class="org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate">  
        <constructor-arg ref="dataSource"/>  
    </bean>  
    <bean id="dataSource"  
          class="org.springframework.jdbc.datasource.DriverManagerDataSource">  
        <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>  
        <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>  
        <property name="username" value="system"/>  
        <property name="password" value="durga"/>  
    </bean>  
</beans>
```

Test.java

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Customer;
import com.durgasoft.dao.CustomerDao;

public class Test {

    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        CustomerDao dao = (CustomerDao)context.getBean("customerDao");
        Customer c = new Customer();
        c.setCid("C-111");
        c.setCname("AAA");
        c.setCaddr("Hyd");
        String status = dao.add(c);
        System.out.println("Student Insertion :" +status);

        Customer c1 = dao.search("C-111");
        System.out.println("Customer Details");
        System.out.println("-----");
        System.out.println("Customer Id : " +c1.getCid());
        System.out.println("Customer Name : " +c1.getCname());
        System.out.println("Customer Address : " +c1.getCaddr());
        System.out.println();
        Customer c2 = new Customer();
        c2.setCid("C-111");
        c2.setCname("BBB");
        c2.setCaddr("Sec");
        status = dao.update(c2);
        System.out.println("Student Updation Status :" +status);
        Customer c3 = dao.search("C-111");
        System.out.println("Customer Updated Details");
        System.out.println("-----");
        System.out.println("Customer Id : " +c3.getCid());
        System.out.println("Customer Name : " +c3.getCname());
        System.out.println("Customer Address : " +c3.getCaddr());
        System.out.println();
        status = dao.delete("C-111");
        System.out.println("Student Deletion Status :" +status);
    }
}
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

SimpleJdbcTemplate:

In Spring JDBC module, the main intention of SimpleJdbcTemplate class is to provide support for JDK5.0 version features like Auto Boxing, Auto Unboxing, Var-Ag methods,.....

SimpleJdbcTemplate class was provided in Spring2.5 version only and it was deprecated in the later versions Spring3.x and Spring4.x , in Spring5.x version SimpleJdbcTemplate class was removed.

If we want to use SimpleJdbcTemplate class we have to use Spring2.5 version jar files in Spring applications.

To execute SQL queries , SimpleJdbcTemplate class has provided the following methods.

public Object execute(String sqlQuery)

Note: To use this method we have to get JdbcOperations class by using getJdbcOperations() method.

public int update(String query, Object ... params)

public Object queryForInt(String query, Object ... params)

public Object queryForLong(String query, Object ... params)

public Object query(String query, Object ... params)

public Object queryForObject(String query, Object ... params)

Note: In case of SimpleJdbcTemplate class, to perform retrieval operations, we have to use "ParameterizedRowMapper" instead of RowMapper interface.

Example:

EmployeeDao.java

```
package com.durgasoft.dao;  
  
import com.durgasoft.beans.Employee;  
  
public interface EmployeeDao {  
    public String add(Employee emp);  
    public Employee search(int eno);  
    public String update(Employee emp);  
    public String delete(int eno);  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

EmployeeDaoImpl.java

```
package com.durgasoft.dao;

import org.springframework.jdbc.core.simple.SimpleJdbcTemplate;

import com.durgasoft.beans.Employee;

public class EmployeeDaoImpl implements EmployeeDao{
    private SimpleJdbcTemplate simpleJdbcTemplate;

    String status = "";
    public void setSimpleJdbcTemplate(SimpleJdbcTemplate simpleJdbcTemplate) {
        this.simpleJdbcTemplate = simpleJdbcTemplate;
    }

    @Override
    public String add(Employee emp) {
        String query = "insert into emp1
values("+emp.getEno()+","+emp.getEname()+","+emp.getEsal()+","+emp.getEaddr()+")";
        simpleJdbcTemplate.getJdbcOperations().execute(query);
        status = "SUCCESS";
        return status;
    }

    @Override
    public Employee search(int eno) {
        String query = "select * from emp1 where eno=?";
        Employee emp = simpleJdbcTemplate.queryForObject(query, new
EmployeeMapper(), eno);

        return emp;
    }

    @Override
    public String update(Employee emp) {
        String query = "update emp1 set ename=?, esal =?, eaddr=? where eno=?";
        simpleJdbcTemplate.update(query, emp.getEname(), emp.getEsal(),
emp.getEaddr(), emp.getEno());
        status = "SUCCESS";
        return status;
    }

    @Override
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public String delete(int eno) {  
    String query = "delete from emp1 where eno = ?";  
    simpleJdbcTemplate.update(query, eno);  
    status = "SUCCESS";  
  
    return status;  
}  
}
```

Employee.java

```
package com.durgasoft.beans;  
  
public class Employee {  
    private int eno;  
    private String ename;  
    private float esal;  
    private String eaddr;  
  
    public int getEno() {  
        return eno;  
    }  
    public void setEno(int eno) {  
        this.eno = eno;  
    }  
    public String getEname() {  
        return ename;  
    }  
    public void setEname(String ename) {  
        this.ename = ename;  
    }  
    public float getEsal() {  
        return esal;  
    }  
    public void setEsal(float esal) {  
        this.esal = esal;  
    }  
    public String getEaddr() {  
        return eaddr;  
    }  
    public void setEaddr(String eaddr) {  
        this.eaddr = eaddr;  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
}
```

EmployeeMapper.java

```
package com.durgasoft.dao;

import java.sql.ResultSet;
import java.sql.SQLException;

import org.springframework.jdbc.core.RowMapper;
import org.springframework.jdbc.core.simple.ParameterizedRowMapper;

import com.durgasoft.beans.Employee;

public class EmployeeMapper implements ParameterizedRowMapper<Employee> {
    @Override
    public Employee mapRow(ResultSet rs, int row_No) throws SQLException {
        Employee emp = new Employee();
        emp.setEno(rs.getInt("ENO"));
        emp.setEname(rs.getString("ENAME"));
        emp.setEsal(rs.getFloat("ESAL"));
        emp.setEaddr(rs.getString("EADDR"));
        return emp;
    }
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd">

    <bean id="employeeDao" class="com.durgasoft.dao.EmployeeDaoImpl">
        <property name="simpleJdbcTemplate" ref="simpleJdbcTemplate"/>
    </bean>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<bean id="simpleJdbcTemplate"
class="org.springframework.jdbc.core.simple.SimpleJdbcTemplate">
    <constructor-arg ref="dataSource"/>
</bean>
<bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
    <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
    <property name="username" value="system"/>
    <property name="password" value="durga"/>
</bean>

</beans>
```

Test.java

```
package com.durgasoft.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Employee;
import com.durgasoft.dao.EmployeeDao;

public class Test {

    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        EmployeeDao dao = (EmployeeDao)context.getBean("employeeDao");
        Employee emp = new Employee();
        emp.setEno(111);
        emp.setEname("AAA");
        emp.setEsal(5000);
        emp.setEaddr("Hyd");
        String status1 = dao.add(emp);
        System.out.println("Employee Insertion Status :" +status1);
        System.out.println();
        Employee emp1 = dao.search(111);
        System.out.println("Employee Details");
        System.out.println("-----");
        System.out.println("Employee Number :" +emp1.getEno());
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
System.out.println("Employee Name :"+emp1.getEname());
System.out.println("Employee Salary :" + emp1.getEsal());
System.out.println("Employee Address :" + emp1.getEaddr());
System.out.println();
Employee emp2 = new Employee();
emp2.setEno(111);
emp2.setEname("BBB");
emp2.setEsal(6000);
emp2.setEaddr("Sec");
String status2 = dao.update(emp2);
System.out.println("Employee Updation Status :" + status2);
System.out.println();
String status3 = dao.delete(111);
System.out.println("Employee Deletion Status :" + status3);
}
```

DAO Support Classes:

In Spring JDBC , we have to prepare DAO implementation classes with XXXTemplate property and the corresponding setXXX() method inorder to inject XXXTemplate class.

In Spring JDBC applications, if we want to get XXXTemplate classes with out declaring Template properties and corresponding setXXX() methods we have to use DAO Support classes provided Spring JDBC module.

There are three types of DAOSupport classes inorder to get Template object in DAO classes.

- 1.JdbcDaoSupport
- 2.NamedParameterJdbcDaoSupport
- 3.SimpleJdbcDaoSupport

Where JdbcDaoSupport class will provide JdbcTemplate reference in DAO classes by using the following method.

```
public JdbcTemplate getJdbcTemplate()
```

Where NamesParameterJdbcDaoSupport class will provide NamedParameterJdbcTemplate reference in DAO classes by using the following method.

```
public NamedParameterJdbcTemplate getNamedparameterJdbctemplate()
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where SimpleJdbcDaoSupport class is able to provide SimpleJdbcTemplate reference in Dao class by using the following method.

```
public SimpleJdbcTemplate getSimpleJdbcTemplate()
```

EX:

```
public class EmployeeDaoImpl extends JdbcDaoSupport implements EmployeeDao{  
  
    public String insert(int eno, String ename, float esal, String eaddr){  
        getJdbcTemplate().update("insert into emp1 values("+eno+","+ename+","+esal+","+eaddr+");");  
        return "SUCCESS";  
    }  
----  
----  
}
```

Spring Batch Updations

Batch Processing:

To perform Batch Updations in Spring JDBC we have to use the following method from JdbcTemplate class.

```
public int[] batchUpdate(String sql_prepared_Statement, BatchPreparedStatementSetter setter)
```

Where BatchPreparedStatementSetter interface contains the following two methods

```
public void setValues(PreparedStatement ps, int index)  
public int getBatchSize()
```

Where setValues() method will be executed for each and every record to set values to the positional parameters existed in PreparedStatement object by getting values from the provided List.

Employee.java

```
package com.durgasoft.beans;  
  
public class Employee {  
    private int eno;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private String ename;
private float esal;
private String eaddr;

public int getEno() {
    return eno;
}
public void setEno(int eno) {
    this.eno = eno;
}
public String getEname() {
    return ename;
}
public void setEname(String ename) {
    this.ename = ename;
}
public float getEsal() {
    return esal;
}
public void setEsal(float esal) {
    this.esal = esal;
}
public String getEaddr() {
    return eaddr;
}
public void setEaddr(String eaddr) {
    this.eaddr = eaddr;
}
```

EmployeeDao.java

```
package com.durgasoft.dao;
import java.util.List;
import com.durgasoft.beans.Employee;

public interface EmployeeDao {
    public int[] insert(List<Employee> list);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

EmployeeDaoImpl.java

```
package com.durgasoft.dao;

import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.util.List;

import javax.sql.DataSource;

import org.springframework.jdbc.core.BatchPreparedStatementSetter;
import org.springframework.jdbc.core.JdbcTemplate;

import com.durgasoft.beans.Employee;

public class EmployeeDaoImpl implements EmployeeDao{
    private DataSource dataSource;
    JdbcTemplate jdbcTemplate;
    public void setDataSource(DataSource dataSource) {
        this.dataSource = dataSource;
        jdbcTemplate = new JdbcTemplate(dataSource);
    }
    @Override
    public int[] insert(List<Employee> list) {
        int[] rowCounts = null;
        try {
            String sql = "insert into emp1 values(?, ?, ?, ?)";
            rowCounts = jdbcTemplate.batchUpdate(sql, new
BatchPreparedStatementSetter() {
                @Override
                public void setValues(PreparedStatement ps, int i) throws
SQLException {
                    ps.setInt(1, list.get(i).getEno());
                    ps.setString(2, list.get(i).getEname());
                    ps.setFloat(3, list.get(i).getEsal());
                    ps.setString(4, list.get(i).getEaddr());
                }
            }
        }
        @Override
        public int getBatchSize() {
            // TODO Auto-generated method stub
        }
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        return list.size();
    }
}
} catch (Exception e) {
    e.printStackTrace();
}
returnrowCounts;
}
```

Jdbc.properties

```
jdbc.driverClassName = oracle.jdbc.OracleDriver
jdbc.url = jdbc:oracle:thin:@localhost:1521:xe
jdbc.username = system
jdbc.password = durga
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd">

    <bean id="empDao" class="com.durgasoft.dao.EmployeeDaolmpl">
        <property name="dataSource" ref="dataSource"/>
    </bean>

    <bean id="dataSource"
          class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="${jdbc.driverClassName}"/>
        <property name="url" value="${jdbc.url}"/>
        <property name="username" value="${jdbc.username}"/>
        <property name="password" value="${jdbc.password}"/>
    </bean>

    <context:property-placeholder location="jdbc.properties"/>
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

</beans>

Test.java

```
package com.durgasoft.test;

import java.util.ArrayList;
import java.util.List;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Employee;
import com.durgasoft.dao.EmployeeDao;

public class Test {

    public static void main(String[] args) {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        EmployeeDao dao = (EmployeeDao)context.getBean("empDao");
        List<Employee> list = new ArrayList<Employee>();
        Employee e1 = new Employee();
        e1.setEno(111);
        e1.setEname("AAA");
        e1.setEsal(5000);
        e1.setEaddr("Hyd");
        list.add(e1);
        Employee e2 = new Employee();
        e2.setEno(222);
        e2.setEname("BBB");
        e2.setEsal(6000);
        e2.setEaddr("Hyd");
        list.add(e2);
        Employee e3 = new Employee();
        e3.setEno(333);
        e3.setEname("CCC");
        e3.setEsal(6000);
        e3.setEaddr("Hyd");
        list.add(e3);

        int[] rowCounts = dao.insert(list);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

        for(int i = 0; i<rowCounts.length; i++) {
            System.out.println(rowCounts[i]);
        }
    }
}

```

Stored Procedure and Functions in Spring JDBC:

If we want to access stored procedures and functions which are available at database from Spring Jdbc application then we have to use "SimpleJdbcCall".

To use SimpleJdbcCall in Spring Jdbc applications we have to use the following steps.

- 1)Create DAO interface and its implementation class.
- 2)IN DAO implementation class, we have to declare DataSource and JdbcTemplate and its respective setter method .
- 3)In side setter method we have to create SimpleJdbcCall object.

```

SimpleJdbcCall jdbcCall = new SimpleJdbcCall();
    jdbcCall.withProcedureName("proc_Name");
4)Configure DataSource and DAO implementation class in beans configuration file.
5)Access "execute" method by passing IN type parameters values in the form of
"SqlParameterSource".
public Map execute(Map m)
public Map execute(SqlParameterSource paramSource)
public Map execute(Object ... obj)

```

Procedure to copy at Database:

```

create or replace procedure getSalary(no IN number, sal OUT
int)
AS
BEGIN
select esal into sal from emp1 where eno = no;
END getSalary;
/

```

Employee.java

```
package com.durgasoft.beans;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public class Employee {  
    private int eno;  
    private String ename;  
    private float esal;  
    private String eaddr;  
  
    public int getEno() {  
        return eno;  
    }  
    public void setEno(int eno) {  
        this.eno = eno;  
    }  
    public String getEname() {  
        return ename;  
    }  
    public void setEname(String ename) {  
        this.ename = ename;  
    }  
    public float getEsal() {  
        return esal;  
    }  
    public void setEsal(float esal) {  
        this.esal = esal;  
    }  
    public String getEaddr() {  
        return eaddr;  
    }  
    public void setEaddr(String eaddr) {  
        this.eaddr = eaddr;  
    }  
}
```

EmployeeDao.java

```
package com.durgasoft.dao;  
  
import java.util.Map;  
  
import com.durgasoft.beans.Employee;  
  
public interface EmployeeDao {
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public void create(Employee emp);
public Object getEmployeeSalary(int eno);
}
```

EmployeeDaoImpl.java

```
package com.durgasoft.dao;

import java.util.Map;

import javax.sql.DataSource;

import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;
import org.springframework.jdbc.core.namedparam.SqlParameterSource;
import org.springframework.jdbc.core.simple.SimpleJdbcCall;

import com.durgasoft.beans.Employee;

public class EmployeeDaoImpl implements EmployeeDao {
private DataSource dataSource;
private SimpleJdbcCall jdbcCall;
public void setDataSource(DataSource dataSource) {
this.dataSource = dataSource;
jdbcCall = new SimpleJdbcCall(dataSource).withProcedureName("getSalary");
}

@Override
public void create(Employee emp) {
try {
JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);
String sql= "insert into emp1 values("+emp.getEno()+","+emp.getEname()+" , "+emp.getEsal()+" , "+emp.getEaddr());
jdbcTemplate.update(sql);
} catch(Exception e) {
e.printStackTrace();
}
}
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
@Override  
public Object getEmployeeSalary(int eno) {  
SqlParameterSource in = new MapSqlParameterSource().addValue("no", eno);  
Map<String, Object> map = jdbcCall.execute(in);  
  
// System.out.println(map);  
return map.get("SAL");  
}  
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
xsi:schemaLocation="  
http://www.springframework.org/schema/beans  
http://www.springframework.org/schema/beans/spring-  
beans.xsd  
http://www.springframework.org/schema/context  
  
http://www.springframework.org/schema/context/spring-  
context.xsd">  
<bean id="empDao" class="com.durgasoft.dao.EmployeeDaoImpl">  
<property name="dataSource" ref="dataSource"/>  
</bean>  
<bean id = "dataSource" class =  
"org.springframework.jdbc.datasource.DriverManagerDataSource">  
<property name = "driverClassName" value = "oracle.jdbc.OracleDriver"/>  
<property name = "url" value = "jdbc:oracle:thin:@localhost:1521:xe"/>  
<property name = "username" value = "system"/>  
<property name = "password" value = "durga"/>  
</bean>  
</beans>
```

Test.java

```
package com.durgasoft.test;  
  
import org.springframework.context.ApplicationContext;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import org.springframework.context.support.ClassPathXmlApplicationContext;  
  
import com.durgasoft.beans.Employee;  
import com.durgasoft.dao.EmployeeDao;  
  
public class Test {  
  
    public static void main(String[] args) throws Exception {  
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
        EmployeeDao dao = (EmployeeDao)context.getBean("empDao");  
  
        Employee emp1 = new Employee();  
        emp1.setEno(111);  
        emp1.setEname("AAA");  
        emp1.setEsal(5000);  
        emp1.setEaddr("Hyd");  
        dao.create(emp1);  
        Object salary1 = dao.getEmployeeSalary(emp1.getEno());  
        System.out.println(emp1.getEno()+"---->"+salary1);  
  
        Employee emp2 = new Employee();  
        emp2.setEno(222);  
        emp2.setEname("BBB");  
        emp2.setEsal(6000);  
        emp2.setEaddr("Hyd");  
        dao.create(emp2);  
        Object salary2 = dao.getEmployeeSalary(emp2.getEno());  
        System.out.println(emp2.getEno()+"---->"+salary2);  
  
        Employee emp3 = new Employee();  
        emp3.setEno(333);  
        emp3.setEname("CCC");  
        emp3.setEsal(7000);  
        emp3.setEaddr("Hyd");  
        dao.create(emp3);  
        Object salary3 = dao.getEmployeeSalary(emp3.getEno());  
        System.out.println(emp3.getEno()+"---->"+salary3);  
    }  
}
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

Using CURSOR Types in Procedures:

Procedure from Spring JDBC Application by using SimpleJdbcCall :

If we want to use CURSOR types in Stored Procedures inorder to retrive multiple Records data then we have to use the following method on SimpleJdbcCall reference.

```
SimpleJdbcCall jdbcCall = new SimpleJdbcCall(dataSource)
    jdbcCall = jdbcCall.withProcedureName("getAllEmployees");
    jdbcCall =
    jdbcCall.returningResultSet("emps",BeanPropertyRowMapper.newInstance(Employee.class));
```

After adding returningResultSet(--,--) method, if we access execute() method on SimpleJdbcCall then execute() method will execute procedure, it will get all the results from CURSOR type variable and stored all recordfs in the form of Employee objects in an ArrayList object with "emps"[CURSOR TYPE variable] key in a Map.

Example:

COPY this Procedure in Database

```
create or replace procedure getAllEmployees(emps OUT SYS_REFCURSOR)
AS
BEGIN
open emps for
select * from emp1;
END getAllEmployees;
/
```

Employee.java

```
package com.durgasoft.beans;

public class Employee {
private int eno;
private String ename;
private float esal;
private String eaddr;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public int getEno() {  
    return eno;  
}  
public void setEno(int eno) {  
    this.eno = eno;  
}  
public String getEname() {  
    return ename;  
}  
public void setEname(String ename) {  
    this.ename = ename;  
}  
public float getEsal() {  
    return esal;  
}  
public void setEsal(float esal) {  
    this.esal = esal;  
}  
public String getEaddr() {  
    return eaddr;  
}  
public void setEaddr(String eaddr) {  
    this.eaddr = eaddr;  
}
```

EmployeeDao.java

```
package com.durgasoft.dao;  
  
import java.util.Map;  
  
public interface EmployeeDao {  
    public Map<String, Object> getAllEmployees();  
}
```

EmployeeDaoImpl.java

```
package com.durgasoft.dao;
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

import java.util.Map;

import javax.sql.DataSource;

import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.simple.SimpleJdbcCall;

import com.durgasoft.beans.Employee;

public class EmployeeDaoImpl implements EmployeeDao {
private SimpleJdbcCall jdbcCall;
private DataSource dataSource;
public void setDataSource(DataSource dataSource) {
this.dataSource = dataSource;
jdbcCall = new SimpleJdbcCall(dataSource).withProcedureName("getAllEmployees");
jdbcCall = jdbcCall.returningResultSet("emps",
BeanPropertyRowMapper.newInstance(Employee.class));
}
@Override
public Map<String, Object> getAllEmployees() {
Map<String, Object> map = jdbcCall.execute();
//System.out.println(map);
return map;
}
}

```

applicationContext.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/s.../context/spring-context.xsd">
<bean id="empDao" class="com.durgasoft.dao.EmployeeDaoImpl">
<property name="dataSource" ref="dataSource"/>
</bean>
<bean id = "dataSource"

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
class = "org.springframework.jdbc.datasource.DriverManagerDataSource">
<property name = "driverClassName" value = "oracle.jdbc.OracleDriver"/>
<property name = "url" value = "jdbc:oracle:thin:@localhost:1521:xe"/>
<property name = "username" value = "system"/>
<property name = "password" value = "durga"/>
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;

import java.util.ArrayList;
import java.util.Map;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Employee;
import com.durgasoft.dao.EmployeeDao;

public class Test {

    public static void main(String[] args) throws Exception {
        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
        EmployeeDao dao = (EmployeeDao)context.getBean("empDao");
        Map<String, Object> map = dao.getAllEmployees();
        System.out.println(map);
        ArrayList<Employee> list =(ArrayList<Employee>) map.get("emps");
        System.out.println(list);

        System.out.println("Employee Details");
        System.out.println("ENO\tENAME\tESAL\tEADDR");
        System.out.println("-----");
        for(Employee e: list) {
            System.out.println(e.getEno()+"\t"+e.getEname()+"\t"+e.getEsal()+"\t"+e.getEaddr());
        }
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Blob and Clob processing in Spring JDBC:

BLOB: It is a data type available at Databases to represent large volumes of binary data.

CLOB: It is a data type available at Database to represent large volumes of character data.

In Spring JDBC Applications, to process BLOB and CLOB Data, Spring JDBC has provided the following three interfaces mainly.

1. AbstractLobCreatingPreparedStatementCallback

--> It will be used to store Blob and Clob related data in Database.

```
protected void setValues(PreparedStatement ps, LobCreator lobCreator) throws SQLException,  
DataAccessException
```

2. AbstractLobStreamingResultSetExtractor

--> It will be used to retrieve BLOB and CLOB data from database.

```
streamData(ResultSet rs) throws SQLException, IOException, DataAccessException
```

3. LobCreator

--> It contains the following methods to prepare Binary stream and character streams to send blob and clob data to database.

```
setBlobAsBinaryStream()  
setClobAsCharacterStream()
```

4. LobHolder

--> It contains the following methods to get Binary stream and character stream to get blob and clob data.

```
getBlobAsBinaryStream()  
getClobAsCharacterStream()
```

Example:

Employee.java

```
package com.durgasoft.beans;  
  
import java.io.File;  
  
public class Employee {  
    private int eno;  
    private String ename;  
    private File emp_Image;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private File emp_Resume;

public int getEno() {
    return eno;
}

public void setEno(int eno) {
    this.eno = eno;
}

public String getEname() {
    return ename;
}

public void setEname(String ename) {
    this.ename = ename;
}

public File getEmp_Image() {
    return emp_Image;
}

public void setEmp_Image(File emp_Image) {
    this.emp_Image = emp_Image;
}

public File getEmp_Resume() {
    return emp_Resume;
}

public void setEmp_Resume(File emp_Resume) {
    this.emp_Resume = emp_Resume;
}
```

EmployeeDao.java

```
package com.durgasoft.dao;

import com.durgasoft.beans.Employee;

public interface EmployeeDao {
    public void insertEmployee(Employee emp);
    public Employee readEmployee();
}
```

EmployeeDaoImpl.java

```
package com.durgasoft.dao;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.sql.DataSource;

import org.springframework.dao.DataAccessViolationException;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.support.AbstractLobCreatingPreparedStatementCallback;
import org.springframework.jdbc.core.support.AbstractLobStreamingResultSetExtractor;
import org.springframework.jdbc.support.lob.LobCreator;
import org.springframework.jdbc.support.lob.LobHandler;
import org.springframework.util.FileCopyUtils;

import com.durgasoft.beans.Employee;

public class EmployeeDaoImpl implements EmployeeDao {
    private LobHandler lobHolder;
    private DataSource dataSource;
    private JdbcTemplate jdbcTemplate;

    public void setDataSource(DataSource dataSource) {
        this.dataSource = dataSource;
        jdbcTemplate = new JdbcTemplate(dataSource);
    }
    public void setLobHolder(LobHandler lobHolder) {
        this.lobHolder = lobHolder;
    }
    public LobHandler getLobHolder() {
        return lobHolder;
    }
    @Override
    public void insertEmployee(Employee emp) {
        String sql_Query = "insert into emp10 values(?, ?, ?, ?)";
        jdbcTemplate.execute(sql_Query, new
AbstractLobCreatingPreparedStatementCallback(lobHolder) {
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

@Override
protected void setValues(PreparedStatement ps, LobCreator lobCreator)
throws SQLException, DataAccessException {
    FileInputStream fis = null;
    FileReader fr = null;
    try {
        ps.setInt(1, emp.getEno());
        ps.setString(2, emp.getEname());
        fis = new FileInputStream(emp.getEmp_Image());
        fr = new FileReader(emp.getEmp_Resume());
        lobCreator.setBlobAsBinaryStream(ps, 3, fis,
(int)emp.getEmp_Image().length());
        lobCreator.setClobAsCharacterStream(ps, 4, fr,
(int)emp.getEmp_Resume().length());
    }catch(IOException e) {
        e.printStackTrace();
    }
}
}

@Override
public Employee readEmployee() {
    Employee emp = new Employee();

    String sql = "select * from emp10";
    jdbcTemplate.query(sql, new
AbstractLobStreamingResultSetExtractor<Object>() {
    @Override
    protected void streamData(ResultSet rs) throws SQLException,
IOException, DataAccessException {
        emp.setEno(rs.getInt(1));
        emp.setEname(rs.getString(2));

        File file1 = new File("E:/spring/my_image.jpg");
        FileOutputStream fos = new FileOutputStream(file1);
        FileCopyUtils.copy(lobHolder.getBlobAsBinaryStream(rs, 3), fos);
        emp.setEmp_Image(file1);

        File file2 = new File("E:/spring/my_resume.docx");
        FileWriter fw = new FileWriter(file2);
        FileCopyUtils.copy(lobHolder.getClobAsCharacterStream(rs, 4), fw);
    }
}
}

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        emp.setEmp_Resume(file2);

    }

}

return emp;
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd">

    <bean id="empDao" class="com.durgasoft.dao.EmployeeDaolmpl">
        <property name="dataSource" ref="dataSource"/>
        <property name="lobHolder" ref="lobHolder"/>
    </bean>

    <bean id="dataSource"
          class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="${jdbc.driverClassName}"/>
        <property name="url" value="${jdbc.url}"/>
        <property name="username" value="${jdbc.username}"/>
        <property name="password" value="${jdbc.password}"/>
    </bean>
    <bean id="lobHolder" class="org.springframework.jdbc.support.lob.DefaultLobHandler">
    </bean>
    <context:property-placeholder location="jdbc.properties"/>
</beans>
```

jdbc.properties

jdbc.driverClassName = oracle.jdbc.OracleDriver

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
jdbc.url = jdbc:oracle:thin:@localhost:1521:xe
jdbc.username = system
jdbc.password = durga
```

Test.java

```
package com.durgasoft.test;

import java.io.File;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.durgasoft.beans.Employee;
import com.durgasoft.dao.EmployeeDao;

public class Test {

    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        EmployeeDao dao = (EmployeeDao)context.getBean("empDao");
        File file1 = new File("E:/spring/nag.jpg");
        File file2 = new File("E:/spring/nag_resume.docx");
        Employee emp1 = new Employee();
        emp1.setEno(111);
        emp1.setEname("Nag");
        emp1.setEmp_Image(file1);
        emp1.setEmp_Resume(file2);
        dao.insertEmployee(emp1);
        System.out.println("Employee Inserted Successfully");

        Employee emp2 = dao.readEmployee();
        System.out.println("Employee Retrived Successfully");
        System.out.println("Employee Details");
        System.out.println("-----");
        System.out.println("Employee Number : "+emp2.getEno());
        System.out.println("Employee Name : "+emp2.getEname());
        System.out.println("Employee Image : "+emp2.getEmp_Image().getAbsolutePath());
        System.out.println("Employee Resume
:"+emp2.getEmp_Resume().getAbsolutePath());
    }
}
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring JDBC Connection Pooling Mechanism

In Database related applications, if we want to perform database operations first we have to create Connection object then we have to close connection object when the database operations are completed.

IN Database related applications, every time creating Connection object and every time destroying Connection object may reduce application performance, because, Creating Connection object and destroying Connection object are two expensive processes, which may reduce application performance.

To overcome the above problem we have to use Connection Pooling in applications. In Connection pooling we will create a set of Connection object in the form of a pool at the application startup time and we will reuse that Connection objects while executing applications , when database operations are completed then we will send back that connection objects to Pool object without destroying that connection objects.

In SPring JDBC applications there are three approaches to provide connection pooling.

- 1.Default Connection Pooling Mech.
- 2.Third Party Connection Pooling Mechanisms
- 3.Application Servers provided Connection Pooling Mechanism

1.Default Connection Pooling Mech.

In SPring Framework, Default Connection pooling mechanism is existed in the form of org.springframework.jdbc.datasource.DriverManagerDataSource, it is useful upto testing only, it is useful for production environment of the application.

If we want to use default Connection Pooling mechanism in SPring JDBC application then we have to configure org.springframework.jdbc.datasource.DriverManagerDataSource in beans configuration file with the following properties .

- 1.driverClassName
- 2.url
- 3.username
- 4.password

EX:

```
<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
    <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<property name="username" value="system"/>
<property name="password" value="durga"/>
</bean>
```

2.Third Party Connection Pooling Mechanisms

In Spring JDBC applications we are able to use the following third party connection pooling mechanisms

- 1.Apache DBCP
- 2.C3P0
- 3.Proxool

1.Apache DBCP:

To use Apcahes DBCP connection pooling mechanism then we have to configure org.apache.commons.dbcp2.BasicDataSource class with the following properties in spring beans configuration file.

- 1.driverClassName
- 2.url
- 3.username
- 4.password
- 5.initialSize:It will take Initial pool size.
- 6.maxTotal: It will allow the specified no of max connections.

EX:

```
<bean id="dataSource" class="org.apache.commons.dbcp2.BasicDataSource">
<property name="driverClassName" value="oracle.jdbc.OracleDriver" />
<property name="url" value="jdbc:oracle:thin:@localhost:1521:xe" />
<property name="username" value="root" />
<property name="password" value="root" />
<property name="initialSize" value="20" />
<property name="maxTotal" value="30" />
</bean>
```

Note: To use this mechanism in Spring JDBC Applications then we have to add the following two jar files to Library.

- 1.commonsd-bcp2-2.2.0.jar
- 2.commonspool2-2.5.0.jar

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2.C3P0

To use C3P0 connection pooling mechanism then we have to configure com.mchange.v2.c3p0.ComboPooledDataSource class with the following properties in spring beans configuration file.

- 1.driverClass
 - 2.jdbcUrl
 - 3.user
 - 4.password
 - 5.minPoolSize: It will take Initial pool size.
 - 6.maxPoolSize: It will allow the specified no of max connections.
 - 7.maxStatements: Max statements it allows.
 - 8.testConnectionOnCheckOut:true/false for Checking Connection before use.
-
-

EX:

```
<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
    <property name="driverClass" value="oracle.jdbc.OracleDriver" />
    <property name="jdbcUrl" value="jdbc:oracle:thin:@localhost:1521:xe" />
    <property name="user" value="system" />
    <property name="password" value="durga" />
    <property name="maxPoolSize" value="30" />
    <property name="minPoolSize" value="10" />
    <property name="maxStatements" value="100" />
    <property name="testConnectionOnCheckout" value="true" />
</bean>
```

Note: To use this mechanism in Spring JDBC Applications then we have to add the following two jar files to Library.

- 1.c3p0-0.9.5.2.jar
- 2.mchange-commons-java-0.2.11.jar

3.Proxool:

To use Proxool connection pooling mechanism then we have to configure org.logicalcobwebs.proxool.ProxoolDataSource class with the following properties in spring beans configuration file.

- 1.driver
- 2.driverUrl
- 3.user

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4.password
5.minimumConnectionCount:It will take Initial pool size.
6.maximumConnectionCount: It will allow the specified no of max connections.

EX:

```
<bean id="dataSource" class="org.logicalcobwebs.proxool.ProxoolDataSource">
    <property name="driver" value="oracle.jdbc.OracleDriver" />
    <property name="driverUrl" value="jdbc:oracle:thin:@localhost:1521:xe" />
    <property name="user" value="system" />
    <property name="password" value="durga" />
    <property name="maximumConnectionCount" value="30" />
    <property name="minimumConnectionCount" value="10" />
</bean>
```

Note: To use this mechanism in Spring JDBC Applications then we have to add the following two jar files to Library.

- 1.proxool-0.9.1.jar
- 2.proxool-cglib.jar

Example:**Employee.java**

```
package com.durgasoft.beans;

public class Employee {
    private int eno;
    private String ename;
    private float esal;
    private String eaddr;

    public int getEno() {
        return eno;
    }
    public void setEno(int eno) {
        this.eno = eno;
    }
    public String getEname() {
        return ename;
    }
    public void setEname(String ename) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
        this.ename = ename;
    }
    public void setEsal(float esal) {
        this.esal = esal;
    }
    public float getEsal() {
        return esal;
    }
    public void setEaddr(String eaddr) {
        this.eaddr = eaddr;
    }
    public String getEaddr() {
        return eaddr;
    }
}
```

EmployeeDao.java

```
package com.durgasoft.dao;

import com.durgasoft.beans.Employee;

public interface EmployeeDao {
    public void insertEmployee(Employee emp);
}
```

EmployeeDaolmpl.java

```
package com.durgasoft.dao;

import javax.sql.DataSource;
import org.springframework.jdbc.core.JdbcTemplate;
import com.durgasoft.beans.Employee;

public class EmployeeDaolmpl implements EmployeeDao {
    private DataSource dataSource;
    private JdbcTemplate jdbcTemplate;

    public void setDataSource(DataSource dataSource) {
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

        this.dataSource = dataSource;
        jdbcTemplate = new JdbcTemplate(dataSource);
    }

    @Override
    public void insertEmployee(Employee emp) {
        String sql_Query = "insert into emp1
values("+emp.getEno()+","+emp.getEname()+","+emp.getEsal()+","+emp.getEaddr()+"";
        jdbcTemplate.execute(sql_Query);
    }
}

```

applicationContext.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd">

    <bean id="empDao" class="com.durgasoft.dao.EmployeeDaolmpl">
        <property name="dataSource" ref="dataSource"/>
    </bean>
    <!-- Default Connection Pooling Mechanism
    <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
        <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
        <property name="username" value="system"/>
        <property name="password" value="durga"/>
    </bean>
    -->
    <!-- DBCP Connection Pooling Mechanism Properties
    <bean id="dataSource" class="org.apache.commons.dbcp2.BasicDataSource">
        <property name="driverClassName" value="oracle.jdbc.OracleDriver" />
        <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe" />
        <property name="username" value="system" />
        <property name="password" value="durga" />
        <property name="initialSize" value="20" />
        <property name="maxTotal" value="30" />
    </bean>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</bean>
-->
<!-- C3P0 Connection Pooling Mechanism Properties
<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
    <property name="driverClass" value="oracle.jdbc.OracleDriver" />
    <property name="jdbcUrl" value="jdbc:oracle:thin:@localhost:1521:xe" />
    <property name="user" value="system" />
    <property name="password" value="durga" />
    <property name="maxPoolSize" value="30" />
    <property name="minPoolSize" value="10" />
    <property name="maxStatements" value="100" />
    <property name="testConnectionOnCheckout" value="true" />
</bean>
-->
<!-- Proxool Connection Pooling Mechanism Properties -->
<bean id="dataSource" class="org.logicalcobwebs.proxool.ProxoolDataSource">
    <property name="driver" value="oracle.jdbc.OracleDriver" />
    <property name="driverUrl" value="jdbc:oracle:thin:@localhost:1521:xe" />
    <property name="user" value="system" />
    <property name="password" value="durga" />
    <property name="maximumConnectionCount" value="30" />
    <property name="minimumConnectionCount" value="10" />
</bean>
</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.durgasoft.beans.Employee;
import com.durgasoft.dao.EmployeeDao;
public class Test {

    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        EmployeeDao dao = (EmployeeDao)context.getBean("empDao");
        Employee emp = new Employee();
        emp.setEno(111);
        emp.setEname("Nag");
        emp.setEsal(5000);
        emp.setEaddr("Hyd");
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28**US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

        dao.insertEmployee(emp);
        System.out.println("Employee Inserted Successfully");

    }
}

```

3. Application Servers provided Connection Pooling Mechanism throw JNDI:

JNDI[Java Naming And Directory Interface]: JNDI is a Middleware Service or an abstraction provided by SUN Microsystems as part of J2EE and which is implemented by all the Application Servers vendors like Weblogic, JBOSS, Glassfish,.....

JNDI is existed inside the application Servers to provide any resource with Global Scope, that is, JNDI will share any resource like "DataSource" to all the applications which are running in the present application server.

In general, almost all the Application Servers are having their own Connection Pooling mechanisms, if we want to use Application Servers provided Connection pooling mechanisms we have to use the following steps.

- 1)Install Application Server.
- 2)Configure Connection Pooling and Datasource in JNDI provided by Application Servers.
- 3)Add the required new JARs to Library.
- 4)Provide JNDI Setups in beans configuration File.

1) Install Application Server[Weblogic Server]

- 1.Download fmw_12.2.1.3.0_wls_quick.jar from internet[oracle.com]
- 2.Open command prompt in Administrator mode.
- 3.Set JAVA8 or JAVA7 in path.
set path=C:\Java\jdk1.8.0_144\bin;
- 4.Goto setup file location and execute JAR file with the following command.
F:\softwares\servers\weblogic>java -jar fmw_12.2.1.3.0_wls_quick.jar
- 5.Click on "Next" button.
- 6.Click on "Next" Button
- 7.Specify Home directory "Oracle"
- 8.Click on "Next" button.
- 9.Click on "Next" button.
- 10.Click on "Next" button
- 11.Click on "Install" button.
- 12.Click on "Next" button.
- 13.Click on "Finish" button.
- 14.Provide domain name "durga_domain".

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 15.Click on "Next" button.
- 16.Click on "Next" button.
- 17.Provide user name and password.
user name: weblogic
password: weblogic_weblogic
confirm password: weblogic_weblogic
- 18.Click on "Next" button.
- 19.Click on "Next" button.
- 20.Select "Adminstration Server"
- 21.CClick on "Next" button.
- 22.Click on "Next" button.
- 23.Click on "Create" button.
- 24.Click on "Next" button.
- 25.Click on "Finish" button.

2)Configure Connection Pooling and Datasource in JNDI provided by Application Servers:

- 1.Goto durga_domain location
C:\Oracle\user_projects\domains\durga_domain
- 2.double click on "startWeblogic" batch file.
- 3.Open Browser and provide the following url to open Administration console.
<http://localhost:7001/console>
- 4.Provide domain user name and password.
user name: weblogic
password: weblogic_weblogic
- 5.Click on "Login" button.
- 6.Go for Domain Structer and select "Services".
- 7.Select "DataSource".
- 8.Click on "New" button.
- 9.Select "Generic datasource".
- 10.Provide the following details.
Name: durgads
Scope: GLOBAL
JNDI Name: durgajndi
Database Type: Oracle
- 11.Click on "Next" button.
- 12.Click on "Next" button.
- 13.Click on "Next" button.
- 14.Provide the following details.
Database Name: xe
Host Name: localhost
Port : 1521
Database User Name: system

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

password: durga
confirm password: durga
15.Click on "Next" button.
16.Click on "Next" button.
17.Select "admin server".
18.Click on "Finish" button.

3)Add the required new JARs to Library:

To use Weblogic Server provided Connection Pooling mechanism in Spring JDBC Application then we have to use the following jars along with the regular jars.

- 1)weblogic.jar
- 2)spring-jdbc-4.0.4.RELEASE.jar
- 3)spring-tx-4.0.4.RELEASE.jar

4)Provide JNDI Setups in beans configure:

To use Weblogic Server provided Connection Pooling mechanism in Spring JDBC Application then we have to provide the following DataSource configuration in beans configuration file.\

```
<bean id="dataSource" class="org.springframework.jndi.JndiObjectFactoryBean">
    <property name="jndiName" value="durgajndi"/>
    <property name="jndiEnvironment">
        <props>
            <prop key="java.naming.factory.initial">weblogic.jndi.WLInitialContextFactory</prop>
            <prop key="java.naming.provider.url">t3://localhost:7001</prop>
        </props>
    </property>
</bean>

<bean id="empDao" class="com.durgasoft.dao.EmployeeDaoImpl">
    <property name="dataSource" ref="dataSource"/>
</bean>
```

Example:

Employee.java

```
package com.durgasoft.beans;
```

```
public class Employee {
    private int eno;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
private String ename;
private float esal;
private String eaddr;

public int getEno() {
    return eno;
}
public void setEno(int eno) {
    this.eno = eno;
}
public String getEname() {
    return ename;
}
public void setEname(String ename) {
    this.ename = ename;
}
public void setEsal(float esal) {
    this.esal = esal;
}
public float getEsal() {
    return esal;
}
public void setEaddr(String eaddr) {
    this.eaddr = eaddr;
}
public String getEaddr() {
    return eaddr;
}
```

EmployeeDao.java

```
package com.durgasoft.dao;

import com.durgasoft.beans.Employee;

public interface EmployeeDao {
    public void insertEmployee(Employee emp);
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EmployeeDaoImpl.java

```
package com.durgasoft.dao;

import javax.sql.DataSource;

import org.springframework.jdbc.core.JdbcTemplate;

import com.durgasoft.beans.Employee;

public class EmployeeDaoImpl implements EmployeeDao {
    private DataSource dataSource;
    private JdbcTemplate jdbcTemplate;

    public void setDataSource(DataSource dataSource) {
        this.dataSource = dataSource;
        jdbcTemplate = new JdbcTemplate(dataSource);
    }
    @Override
    public void insertEmployee(Employee emp) {
        String sql_Query = "insert into emp1
values("+emp.getEno()+","+emp.getEname()+","+emp.getEsal()+","+emp.getEaddr()+")";
        jdbcTemplate.execute(sql_Query);
    }
}
```

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context.xsd">

    <bean id="dataSource" class="org.springframework.jndi.JndiObjectFactoryBean">
        <property name="jndiName" value="durgajndi"/>
        <property name="jndiEnvironment">
```

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28** US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<props>
    <prop key="java.naming.factory.initial">weblogic.jndi.WLInitialContextFactory</prop>
    <prop key="java.naming.provider.url">t3://localhost:7001</prop>
</props>
</property>
</bean>

<bean id="empDao" class="com.durgasoft.dao.EmployeeDaoImpl">
    <property name="dataSource" ref="dataSource"/>
</bean>

</beans>
```

Test.java

```
package com.durgasoft.test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.remoting.rmi.JndiRmiProxyFactoryBean;

import com.durgasoft.beans.Employee;
import com.durgasoft.dao.EmployeeDao;

public class Test {

    public static void main(String[] args) throws Exception {
        ApplicationContext context = new
ClassPathXmlApplicationContext("applicationContext.xml");
        EmployeeDao dao = (EmployeeDao)context.getBean("empDao");
        Employee emp = new Employee();
        emp.setEno(111);
        emp.setEname("AAA");
        emp.setEsal(5000);
        emp.setEaddr("Hyd");
        dao.insertEmployee(emp);
        System.out.println("Employee Inserted Successfully");
    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGA SOFT { SPRING AOP }

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING AOP MODULE INDEX

1.Intorduction.....	PAGE 316
2. AOP Terminology.....	PAGE 318
3. Advices In Spring.....	PAGE 320
4. PointCut.....	PAGE 323
5. AspectJ.....	PAGE 342

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring-AOP

Introduction:

Aspect Orientation:

In general, in enterprise applications development, if we use Object Oriented Programming languages then we have to provide the implementation by combining both Applications business logic and services logic.

EX:

```

1. public class Transaction{
2.   public void deposit(...){
3.     ----Deposit Logic-----
4.     ----Authentocation-----
5.     ----Logging-----
6.     ----Transact-----
7.   }
8.   public void withdraw(...){
9.     ----Deposit Logic-----
10.    ----Authentication-----
11.    ----Logging-----
12.    ----Transact-----
13.  }
14.  public void transfer(...){
15.    ----Deposit Logic-----
16.    ----Authentication-----
17.    ----Logging-----
18.    ----Transact-----
19.  }
20. }
```

If we want to prepare applications in object Orientations style there we have to provide application logic by combining services logics and Business logic as a single Unit. Here Authentications ,Authorization ,logging ,Data Validations,...are services and Client Exact requirement is business logic.

Common Task fo every application we can called as service logic . And actual Business Requirements we called as a businesss logic.

If we use the above style of implementation then we are able to get the following problems.

1. It is very difficult to modify the services logic, it required to modify in all business method.
2. It will not provide Sharability
3. It will not provide Code Reusability.
4. It will provide tightly coupled design.

To overcome the above problems we have to use Aspect Orientation.

Aspect Orientation is not a programming language, it is a methodology or a paradigm, it will be applied on Object Oriented Programming in order to get loosely coupled design and in order to improve sharability and Reusability.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

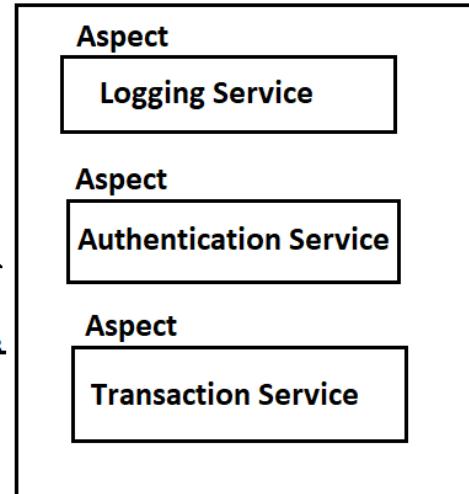
FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

The basic idea behind Aspect Orientation is to separate all services logic from System Business logic , declaring each and every service as an aspect and injecting these aspects into the application Business logic in the respective locations by using Dependence Injection.

```
class Bank_Transaction
{
    public void deposit(){}
    ----Deposit Logic-----
    {
        Injecting Services
        in deposit
    }
    public void withdraw(){}
    ----Withdraw Logic-----
    {
        Injecting Services
        to withdraw
    }
    public void transfer(){}
    ----Transfer Logic -----
    {
        Injecting Services
        to transfer
    }
}
```



In enterprise Applications, AOP will provide the following advantages.

1. In Enterprise applications, Business components looks very clean and having only business implementation statements.
2. All Services are implemented in a common place which simplifies code maintenance.
3. We can make changes in common location , so that, the changes are reflected to all business methods.

AOP is implemented by the following vendors.

1. AspectJ
2. Spring AOP
3. JBOSS AOP

Spring Framework is providing the following two types of implementations

1. Schema Based Implementation
2. AspectJ
 - a. Declarative Based Approach
 - b. Annotation Based Approach

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

AOP Terminology

In general, AOP will use the following terminologies in order to implement AOP based implementation.

1. Aspect
2. Advice
3. Join point
4. Pointcut
5. Target
6. Proxy
7. Weaving
8. Introduction

1. Aspect:

An Aspect is the concern or a service which we want to implement in the application such as logging, transactional , Security etc. *Aspect is name of service.*

2. Advice:

An Advice is the actual implementation of the aspect. Aspect is a concept and Advice is the concrete implementation of the concept.

3. Join point:

JoinPoint is a location where aspect can be applied.

A JoinPoint is a point in the execution of the program where an aspect can be applied. It could be before/after executing the method, before throwing an exception, before/after modifying an instance variable etc.

4. Pointcut:

PointCuts tell on which join points the aspect will be applied. An advice is associated with a point cut expression and is applied to a join point which matches the point cut expression.

5.Target:

Target is a business component class which is being advised

6.Proxy:

Proxy is the object which is created by the framework after applying the advice on the target object.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

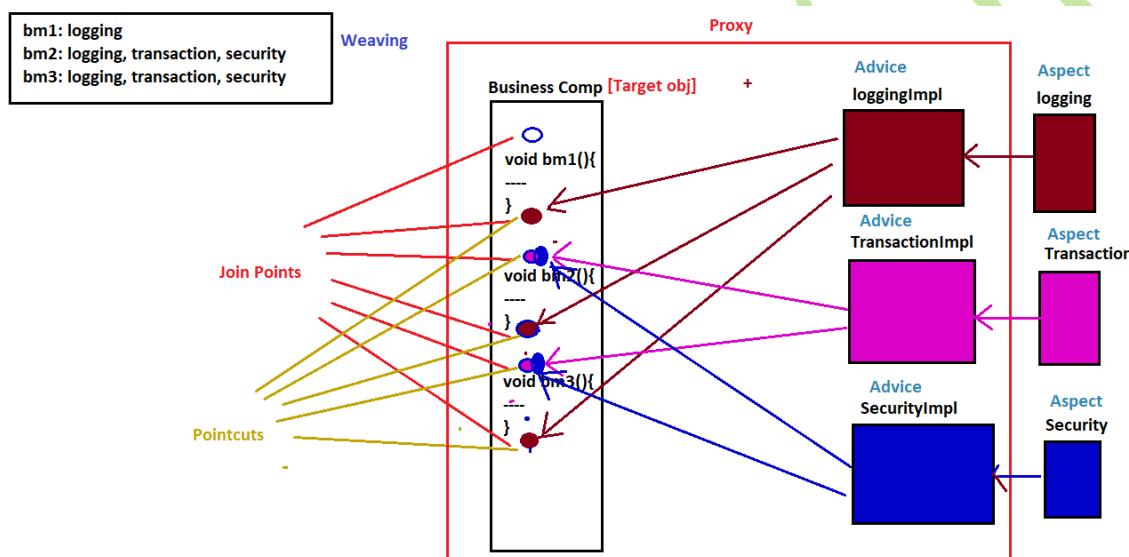
Proxy = target + advice(s)

7. Weaving:

Weaving is the process of applying the aspect on the target object to produce the proxy object. Weaving can be done at compile time, class loading time or runtime. Spring AOP supports weaving at runtime.

8. Introduction:

An Introduction allows adding new methods or attributes to existing classes. The new method and instance variable can be introduced to existing classes without having to change them, giving them new state and behavior.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Advices In Spring

Advices In Spring:

Advice is the implementation of Aspect. An Advice provides the code for implementation of the service or Aspect. As an example consider logging service, logging is an Aspect and Advice denotes the implementation of Log4j.

In general, all the advices code will not be included in business methods at compile time, these services will be included at runtime.

Spring Framework is providing the following various advices.

1. Before Advice
2. After Advice
3. After-Returning
4. After-throwing
5. Around Advice

1. Before Advice:

Before advice contains service/Aspect implementation , it will be executed before executing the respective business method.

To represent Before Advice, Spring Framework has provided a predefined interface in the form of "org.springframework.aop.MethodBeforeAdvice".

If we want to use Before Advice in Spring applications , first, we have to declare an user defined class, it must implement org.springframework.aop.MethodBeforeAdvice interface and we must provide implementation for the following method provided by MethodBeforeAdvice interface.

public void before(Method m, Object[] Bus_Meth_Params, Object target) throws Exception

- ✓ Where java.lang.reflect.Method parameter is able to provide metadata of the Business method to which BeforeAdvice is applied.
- ✓ Where Object[] is providing business method parameter values in the form of Objects.
- ✓ Where Object is representing the target Object.

Note: The services which are implemented in before() method are executed at before executing business logic.

Example:

```
1. public class BeforeAdviceImpl implements MethodBeforeAdvice{
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. public void before(Method method, Object[] params, Object target)throws Exception{
3. -----Service implementation-----
4. }
5. }
```

2.3. After Advice[After Returning Advice] :

It is same as Before Advice, But this advice contains services which are applied after completion of our business method logic

To create an after returning advice in spring, we have to declare an user defined class, it must implement org.springframework.aop.AfterReturningAdvice and we must implement the following method.

```
public void afterReturning(Object returnValue, Object[] args, Object target) throws Exception
```

- ✓ Where first parameter is representing return value in the form of Object type.
- ✓ Where second parameter is able to represent business method parameters in the form of Object[].
- ✓ Where third parameter is representing Target Object.

EX:

```

1. public class AfterReturningAdviceImpl implements AfterReturningAdvice{
2. public void afterReturning(Object returnValue, Object[] args, Object target)throws Exception{
3. ----
4. }
5. }
```

Note: In Schema Based implementation, After Advice and After-Returning Advice are same, but, in Annotation approach both are different.

4 . After-throwing or Throws Advice:

This advice will be executed after throwing an exception from the business method.

To represent After-Throwing Advice, Spring Framework has provided a predefined interface in the form of "org.springframework.aop.ThrowsAdvice".

If we want to use After Throwing Advice in Spring applications , first, we have to declare an user defined class, it must implement org.springframework.aop.ThrowsAdvice interface and we must provide implementation for the following method provided by ThrowsAdvice interface.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public void afterThrowing([Method, args, target], ThrowableSubclass)
```

Some Examples of the above form are

```
public void afterThrowing(Exception ex)
public void afterThrowing(RemoteException)
```

```
public void afterThrowing(Method method, Object[] args, Object target, Exception ex)
public void afterThrowing(Method method, Object[] args, Object target, ServletException ex)
```

- ✓ Where `java.lang.reflect.Method` parameter is able to provide metadata of the Business method to which `BeforeAdvice` is applied.
- ✓ Where `Object[]` is providing business method parameter values in the form of Objects.
- ✓ Where `Object` is representing the target Object.
- ✓ Where `Exception` is representing the generated Exception.

Example:

```
1. public class ThrowsAdviceImpl implements ThrowsAdvice{
2.   public void afterThrows(Method method, Object[] params, Object target) throws Exception
3.   {
4.     -----Service implementation-----
5.   }
}
```

5. Around Advice

- ⊕ Around Advice will be executed before and after executing the business method.
- ⊕ Around Advice is combination of both Before and After Advice.
- ⊕ Around Advice is not given by spring framework and it is from Open Source implementation called AOP alliance.
- ⊕ Around Advice can be used by any framework which supports AOP.

To represent Around Advice, Spring AOP Alliance has provided a predefined interface in the form of `org.aopalliance.intercept.MethodInterceptor`.

`MethodINterceptor` has provided the following method inorder to provide services before and after execution of the business method.

```
public Object invoke(MethodInvocation mi) throws Throwable
```

In Around Advice, we will implement Before and After Advice in `invoke()` method, in `invoke()` method will provide before advice logic before calling `proceed()` method and we will provide After Advice logic after calling `proceed()` method.

Note: Around Advice can access the return value of business method and it can modify the value

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

and it can return a different value back to the client, as return type is Object, but in the After Advice its not possible right, as its return type is void.

Example:

```

1. public class AroundAdviceImpl implements MethodInterceptor
2. {
3.     public Object invoke(MethodInvocation mi)throws Throwable
4.     {
5.         //Before Logic
6.         Object ob = mi.proceed();
7.         //After logic
8.         return ob;
9.     }
10.}
```

PointCut

PointCut defines at what Joinpoints Advices has to be applied, instead of defining advices at all join points.

If we want to use PointCuts in AOP based applications then we have to configure that pointcuts in Spring configuration File.

To configure Pointcuts in configuration file then we have to use the following two types of PointCuts.

1. Static Pointcut
2. Dynamic Pointcut

1. Static Pointcut:

Static pointcuts define advice that is always executed. Static pointcuts are based on method and target class, and cannot take into account the method's arguments. Static pointcuts are sufficient - and best - for most usages. It's possible for Spring to evaluate a static pointcut only once, when a method is first invoked: after that, there is no need to evaluate the pointcut again with each method invocation.

To represent Pointcuts , Spring framework has provided a predefined interface in the form of "org.springframework.aop.PointCut".

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SpringFramework has provided the following Implementation classes for org.springframework.aop.PointCut interface.

1. NameMatchMethodPointcut
2. Perl5RegexpMethodPointcut
3. JdkRegexpMethodPointcut

2. Dynamic Pointcut:

Dynamic pointcuts determine if advice should be executed by examining the runtime method arguments.

Dynamic pointcuts are costlier to evaluate than static pointcuts. They take into account method arguments, as well as static information. This means that they must be evaluated with every method invocation; the result cannot be cached, as arguments will vary.

To represent Dynamic Pointcut Spring has provided the following predefined class.

1. ControlFlowPointcut
2. DynamicMethodMatcherPointcut

If we want to use Pointcuts in Spring applications then we have to configure Pointcut and Advisor in a spring configuration file.

In spring applications, we will use "DefaultPointCutAdvisor" in order to suggest the advices to the Pointcuts.

Where if we want to use NameMatchMethodPointcut then we have to use "mappedNames" property of type "array" and we must provide business method names as values to which we want to apply advices.

EX:

```
1. <bean id="pointcut" class="org.springframework.aop.support.NameMatchMethodPointcut">
2.   <property name="mappedNames">
3.     <array>
4.       <value>displayEmployee</value>
5.       <value>getEmployeeDetails</value>
6.     </array>
7.   </property>
8. </bean>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ Where If we want to use "Perl5RegexpMethodPointcut" and "JdkRegexpMethodPointcut" in spring applications then we have to use a property like "pattern" of list type with different patterns.

EX:

```

1. <bean id="pointcut" class="org.springframework.aop.support.Perl5RegexpMethodPointcut">
2.   <property name="patterns">
3.     <list>
4.       <value>.*Employee.*</value>
5.     </list>
6.   </property>
7. </bean>
```

- ✓ Where if we want to use DefaultPointcutAdvisor in spring applications then we have to use "pointcut" and "advice" properties.

EX:

```

1. <bean id="advisor" class="org.springframework.aop.support.DefaultPointcutAdvisor">
2.   <property name="pointcut" ref="pointcut"/>
3.   <property name="advice" ref="validatorAdvice"/>
4. </bean>
```

Example On Before Advice:

Employee.java

```

1. package com.durgasoft.beans;
2.
3. public class Employee {
4.   private int eno;
5.   private String ename;
6.   private float esal;
7.   private String eemail;
8.   private String emobile;
9.
10.  public int getEno() {
11.    return eno;
12.  }
13.  public void setEno(int eno) {
14.    this.eno = eno;
15.  }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

16. public String getEname() {
17.     return ename;
18. }
19. public void setEname(String ename) {
20.     this.ename = ename;
21. }
22. public float getEsal() {
23.     return esal;
24. }
25. public void setEsal(float esal) {
26.     this.esal = esal;
27. }
28. public String getEemail() {
29.     return eemail;
30. }
31. public void setEemail(String eemail) {
32.     this.eemail = eemail;
33. }
34. public String getEmobile() {
35.     return emobile;
36. }
37. public void setEmobile(String emobile) {
38.     this.emobile = emobile;
39. }
40.
41.
42.}
```

EmployeeService.java

```

1. package com.durgasoft.bo;
2.
3. import com.durgasoft.beans.Employee;
4.
5. public interface EmployeeService {
6.     public void displayEmployee(Employee emp);
7.     public void getEmployeeDetails(Employee emp);
8. }
```

EmployeeServiceImpl.java

```

1. package com.durgasoft.bo;
2.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. import com.durgasoft.beans.Employee;
4.
5. public class EmployeeServiceImpl implements EmployeeService {
6.
7.     @Override
8.     public void displayEmployee(Employee emp) {
9.         System.out.println("Employee Details from displayEmployee---");
10.        System.out.println("-----");
11.        System.out.println("Employee Number : "+emp.getEno());
12.        System.out.println("Employee Name : "+emp.getEname());
13.        System.out.println("Employee Salary : "+emp.getEsal());
14.        System.out.println("Employee Email Id : "+emp.getEmail());
15.        System.out.println("Employee Mobile No : "+emp.getEmobile());
16.
17.    }
18.    public void getEmployeeDetails(Employee emp) {
19.        System.out.println("Employee Details from getEmployeeDetails--");
20.        System.out.println("-----");
21.        System.out.println("Employee Number : "+emp.getEno());
22.        System.out.println("Employee Name : "+emp.getEname());
23.        System.out.println("Employee Salary : "+emp.getEsal());
24.        System.out.println("Employee Email Id : "+emp.getEmail());
25.        System.out.println("Employee Mobile No : "+emp.getEmobile());
26.
27.    }
28.
29.

```

EmployeeValidator.java

```

1. package com.durgasoft.advice;
2.
3. import java.lang.reflect.Method;
4.
5. import org.springframework.aop.MethodBeforeAdvice;
6.
7.
8. import com.durgasoft.beans.Employee;
9.
10. public class EmployeeValidator implements MethodBeforeAdvice {
11.
12.     @Override

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

13. public void before(Method method, Object[] params, Object target) throws Throwable {
14.     Employee emp = (Employee) params[0];
15.     System.out.println("Validation Messages for " + method.getName());
16.     System.out.println("-----");
17.     if( emp.getEno() < 100 || emp.getEno() > 999 ) {
18.         System.out.println("***** Employee Number must be 3 digit number *****");
19.     }
20.     if( emp.getEsal() < 20000 || emp.getEsal() > 50000 ) {
21.         System.out.println("***** Employee Salary Must be in between 20000 to 50000 * * * * *");
22.     }
23.     if(!emp.getEemail().endsWith("@durgasoft.com")) {
24.         System.out.println("***** Employee Email is Invalid *****");
25.     }
26.     if(!emp.getEmobile().startsWith("91-")) {
27.         System.out.println("***** Employee Mobile Number is INValid *****");
28.     }
29. }
30.

```

ApplicationContext.java

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:aop="http://www.springframework.org/schema/aop"
5.   xsi:schemaLocation="
6.       http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
7.       http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">
8.     <!-- Bean Object -->
9.     <bean id="empBean" class="com.durgasoft.beans.Employee">
10.       <property name="eno" value="12345"/>
11.       <property name="ename" value="AAA"/>
12.       <property name="esal" value="10000"/>
13.       <property name="eemail" value="aaa@gmail.com"/>
14.       <property name="emobile" value="9988776655"/>
15.     </bean>
16.
17.     <!-- Target Object -->
18.     <bean id="empService" class="com.durgasoft.bo.EmployeeServiceImpl"/>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

19. 19. <!-- Advice -->
20. 20. <bean id="validatorAdvice" class="com.durgasoft.advice.EmployeeValidator"/>
21.
22.
23. 23. <!-- Pointcut -->
24. 24. <bean id="pointcut" class="org.springframework.aop.support.NameMatchMethodPointc
   ut">
25. 25.   <property name="mappedNames">
26. 26.     <array>
27. 27.       <value>displayEmployee</value>
28. 28.       <value>getEmployeeDetails</value>
29. 29.     </array>
30. 30.   </property>
31. 31. </bean>
32.
33. 33. <!-- Advisor -->
34. 34. <bean id="advisor" class="org.springframework.aop.support.DefaultPointcutAdvisor">
35. 35.   <property name="pointcut" ref="pointcut"/>
36. 36.   <property name="advice" ref="validatorAdvice"/>
37.
38. 38. </bean>
39.
40. 40. <!-- Proxy Object -->
41. 41. <bean id="empProxy" class = "org.springframework.aop.framework.ProxyFactoryBean"
   >
42. 42.   <property name="target" ref="empService"/>
43. 43.   <property name="interceptorNames">
44. 44.     <list>
45. 45.       <value>advisor</value>
46. 46.     </list>
47. 47.   </property>
48. 48. </bean>
49.</beans>

```


Test.java

```

1. package com.durgasoft.test;
2.
3. import org.springframework.aop.support.DefaultPointcutAdvisor;
4. import org.springframework.aop.support.NameMatchMethodPointcut;
5. import org.springframework.context.ApplicationContext;
6. import org.springframework.context.support.ClassPathXmlApplicationContext;
7.

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. import com.durgasoft.beans.Employee;
9. import com.durgasoft.bo.EmployeeService;
10.
11. public class Test {
12.
13.     public static void main(String[] args) {
14.         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
15.         Employee emp = (Employee) context.getBean("empBean");
16.         EmployeeService empService = (EmployeeService) context.getBean("empProxy");
17.         empService.displayEmployee(emp);
18.         System.out.println();
19.         empService.getEmployeeDetails(emp);
20.
21.
22.     }
23. }
```

Example On After Advice:

Student.java

```
1. package com.durgasoft.beans;
2.
3. public class Student {
4.     private String sname;
5.     private String squal;
6.     private String semail;
7.     private String smobile;
8.
9.     public String getSname() {
10.         return sname;
11.     }
12.     public void setSname(String sname) {
13.         this.sname = sname;
14.     }
15.     public String getSqual() {
16.         return squal;
17.     }
18.     public void setSqual(String squal) {
19.         this.squal = squal;
20.     }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

21. public String getSemail() {
22.     return semail;
23. }
24. public void setSemail(String semail) {
25.     this.semail = semail;
26. }
27. public String getSmobile() {
28.     return smobile;
29. }
30. public void setSmobile(String smobile) {
31.     this.smobile = smobile;
32. }
33.
34.
35.}
```

InstituteService.java

```

1. package com.durgasoft.bo;
2.
3. import com.durgasoft.beans.Student;
4.
5. public interface InstituteService {
6.     public void enquiry(Student std, String course_Name);
7.     public void registration(Student std, String course_name);
8. }
```

InstituteserviceImpl.java

```

1. package com.durgasoft.bo;
2.
3. import com.durgasoft.beans.Student;
4.
5. public class InstituteserviceImpl implements InstituteService {
6.
7.     @Override
8.     public void enquiry(Student std, String course_Name) {
9.         System.out.println("Student Enquiry Details");
10.        System.out.println("-----");
11.        System.out.println("Student Name      :" + std.getSname());
12.        System.out.println("Student Qualification :" + std.getQual());
13.        System.out.println("Student Email ID   :" + std.getSEmail());
14.        System.out.println("Student Mobile Number :" + std.getSmobile());
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

15.     System.out.println("Enquiry Course Name : "+course_Name);
16. }
17. public void registration(Student std, String course_Name) {
18.     System.out.println("Student Course Registration Details");
19.     System.out.println("-----");
20.     System.out.println("Student Name : "+std.getSname());
21.     System.out.println("Student Qualification : "+std.getQual());
22.     System.out.println("Student Email ID : "+std.getEmail());
23.     System.out.println("Student Mobile Number : "+std.getMobile());
24.     System.out.println("Enquiry Course Name : "+course_Name);
25. }
26.

```

ThanqAdvice.java

```

1. package com.durgasoft.advice;
2.
3. import java.lang.reflect.Method;
4.
5. import org.springframework.aop.AfterReturningAdvice;
6.
7. import com.durgasoft.beans.Student;
8.
9. public class ThanqAdvice implements AfterReturningAdvice {
10.
11.     @Override
12.     public void afterReturning(Object return_Val, Method method, Object[] params, Object t
    arget) throws Throwable {
13.         Student std = (Student)params[0];
14.         String course_Name = (String)params[1];
15.         System.out.println("ThanQ "+std.getSname()+" for your course "+method.getName()+""
    on "+course_Name);
16.         System.out.println("Durgasoft Team will contact with you for the Course Schedule");
17.     }
18.

```

ApplicationContext.java

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:aop="http://www.springframework.org/schema/aop"
5.     xsi:schemaLocation="

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6.      http://www.springframework.org/schema/beans http://www.springframework.org/sche
ma/beans/spring-beans.xsd
7.      http://www.springframework.org/schema/aop http://www.springframework.org/schema/
aop/spring-aop.xsd">
8.
9.      <!-- Beans -->
10.     <bean id="stdBean" class="com.durgasoft.beans.Student">
11.       <property name="sname" value="Durga"/>
12.       <property name="squal" value="BTech"/>
13.       <property name="semail" value="durga@gmail.com"/>
14.       <property name="smobile" value="91-9988776655"/>
15.     </bean>
16.
17.     <!-- Target -->
18.     <bean id="target" class="com.durgasoft.bo.InstituteServiceImpl"/>
19.
20.     <!-- Advice -->
21.     <bean id="advice" class="com.durgasoft.advice.ThanqAdvice"/>
22.
23.     <!-- Proxy -->
24.     <bean id="proxy" class="org.springframework.aop.framework.ProxyFactoryBean">
25.       <property name="target" ref="target"/>
26.       <property name="interceptorNames">
27.         <list>
28.           <value>advice</value>
29.         </list>
30.       </property>
31.     </bean>
32.</beans>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5.
6. import com.durgasoft.beans.Student;
7. import com.durgasoft.bo.InstituteService;
8.
9. public class Test {
10.   public static void main(String[] args) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11. ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
12. Student std = (Student) context.getBean("stdBean");
13. InstituteService inst_Service = (InstituteService) context.getBean("proxy");
14. inst_Service.enquiry(std, "JAVA");
15. System.out.println();
16. inst_Service.registration(std, "JAVA");
17.
18.
19. }
20.}
```

Example On Throws Advice:

Movie.java



```
1. package com.durgasoft.beans;
2.
3. public class Movie {
4.     private String movie_Name;
5.     private String show_Time;
6.     private int price;
7.
8.     public String getMovie_Name() {
9.         return movie_Name;
10.    }
11.    public void setMovie_Name(String movie_Name) {
12.        this.movie_Name = movie_Name;
13.    }
14.    public String getShow_Time() {
15.        return show_Time;
16.    }
17.    public void setShow_Time(String show_Time) {
18.        this.show_Time = show_Time;
19.    }
20.    public int getPrice() {
21.        return price;
22.    }
23.    public void setPrice(int price) {
24.        this.price = price;
25.    }
26.}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

27.
28.}**MovieService.java**

```
1. package com.durgasoft.bo;
2.
3. import com.durgasoft.beans.Movie;
4.
5. public interface MovieService {
6.     public void playMovie(Movie movie) throws Exception;
7. }
```

MovieServiceImpl.java

```
1. package com.durgasoft.bo;
2.
3. import com.durgasoft.beans.Movie;
4.
5. public class MovieServiceImpl implements MovieService {
6.
7.     @Override
8.     public void playMovie(Movie movie) throws Exception {
9.         System.out.println("Movie Details");
10.        System.out.println("-----");
11.        System.out.println("Movie Name :" + movie.getMovie_Name());
12.        System.out.println("Movie Time :" + movie.getShow_Time());
13.        System.out.println("Price   :" + movie.getPrice());
14.        throw new RuntimeException("Power Failure Occurred");
15.    }
16.
17. }
```

MoneyReturnAdvice.java

```
1. package com.durgasoft.advice;
2.
3.
4. import java.lang.reflect.Method;
5.
6. import org.springframework.aop.ThrowsAdvice;
7.
8.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. public class MoneyReturnAdvice implements ThrowsAdvice {  
10.    public void afterThrowing(Method method, Object[] params, Object target, Exception e)  
11.    {  
12.        System.out.println("Power Failure Exception Occurred, Movie was stopped, please come to counter and collect your money");  
13.    }  
14.}
```



ApplicationContext.java

```
1. <?xml version="1.0" encoding="UTF-8"?>  
2. <beans xmlns="http://www.springframework.org/schema/beans"  
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
4.   xmlns:aop="http://www.springframework.org/schema/aop"  
5.   xsi:schemaLocation="  
6.       http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd  
7.       http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">  
8.     <!-- Beans -->  
9.     <bean id = "movieBean" class="com.durgasoft.beans.Movie">  
10.        <property name="movie_Name" value="Bahubali"/>  
11.        <property name="show_Time" value="6:00pm"/>  
12.        <property name="price" value="250"/>  
13.    </bean>  
14.  
15.     <!-- Target -->  
16.     <bean id="target" class="com.durgasoft.bo.MovieServiceImpl"/>  
17.  
18.     <!-- Advice -->  
19.     <bean id="advice" class="com.durgasoft.advice.MoneyReturnAdvice"/>  
20.  
21.     <!-- Proxy -->  
22.     <bean id="proxy" class="org.springframework.aop.framework.ProxyFactoryBean">  
23.        <property name="target" ref="target"/>  
24.        <property name="interceptorNames">  
25.            <list>  
26.                <value>advice</value>  
27.            </list>  
28.        </property>  
29.    </bean>  
30.</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5.
6. import com.durgasoft.beans.Movie;
7. import com.durgasoft.bo.MovieService;
8.
9. public class Test {
10.
11.     public static void main(String[] args) {
12.         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
13.         Movie movie = (Movie)context.getBean("movieBean");
14.
15.         MovieService movie_Service = (MovieService) context.getBean("proxy");
16.         try {
17.             movie_Service.playMovie(movie);
18.         } catch (Exception e) {
19.
20.         }
21.
22.     }
23.
24. }
```

Example on Around Advice:**Account.java**

```
1. package com.durgasoft.beans;
2.
3. public class Account {
4.     private String accNo;
5.     private String accName;
6.     private String accType;
7.     private int balance;
8.
9.     public String getAccNo() {
10.         return accNo;
11.     }
12. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
12. public void setAccNo(String accNo) {  
13.     this.accNo = accNo;  
14. }  
15. public String getAccName() {  
16.     return accName;  
17. }  
18. public void setAccName(String accName) {  
19.     this.accName = accName;  
20. }  
21. public String getAccType() {  
22.     return accType;  
23. }  
24. public void setAccType(String accType) {  
25.     this.accType = accType;  
26. }  
27. public int getBalance() {  
28.     return balance;  
29. }  
30. public void setBalance(int balance) {  
31.     this.balance = balance;  
32. }  
33.  
34.  
35.}
```

Cheque.java

```
1. package com.durgasoft.beans;  
2.  
3. public class Cheque {  
4.     private String cheque_No;  
5.     private int amount;  
6.  
7.     public String getCheque_No() {  
8.         return cheque_No;  
9.     }  
10.    public void setCheque_No(String cheque_No) {  
11.        this.cheque_No = cheque_No;  
12.    }  
13.    public int getAmount() {  
14.        return amount;  
15.    }  
16.    public void setAmount(int amount) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
17.     this.amount = amount;
18. }
19.}
```

TransactionService.java

```
1. package com.durgasoft.bo;
2.
3. import com.durgasoft.beans.Account;
4. import com.durgasoft.beans.Cheque;
5.
6. public interface TransactionService {
7.     public void debit(Account acc, Cheque cheque );
8. }
```

TransactionServiceImpl.java

```
1. package com.durgasoft.bo;
2.
3. import com.durgasoft.beans.Account;
4. import com.durgasoft.beans.Cheque;
5.
6. public class TransactionServiceImpl implements TransactionService {
7.
8.     @Override
9.     public void debit(Account acc, Cheque cheque) {
10.         int initial_Amount = acc.getBalance();
11.         int debit_Amount = cheque.getAmount();
12.         int total_Amount = initial_Amount - debit_Amount;
13.         acc.setBalance(total_Amount);
14.         System.out.println("*****Transaction Success*****");
15.         System.out.println("*****Amount is debited from Account*****");
16.     }
17. }
```

ChequeClearenceAdvice.java

```
1. package com.durgasoft.advice;
2.
3. import org.aopalliance.intercept.MethodInterceptor;
4. import org.aopalliance.intercept.MethodInvocation;
5.
6.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

7. import com.durgasoft.beans.Account;
8. import com.durgasoft.beans.Cheque;
9.
10. public class ChequeClearenceAdvice implements MethodInterceptor {
11.
12.     @Override
13.     public Object invoke(MethodInvocation mi) throws Throwable {
14.
15.         Object[] params = mi.getArguments();
16.         Account acc = (Account)params[0];
17.         Cheque cheque = (Cheque)params[1];
18.
19.         System.out.println("Hello Customer!, Check No "+cheque.getCheque_No()+" is coming for clearance");
20.         mi.proceed();
21.         System.out.println("Hello Customer!, Account Number "+acc.getAccNo()+" has been debited the amount "+cheque.getAmount()+" in clearence of the cheque No "+cheque.getCheque_No()+" , Now the total Amount in your Account is "+acc.getBalance() );
22.         return null;
23.     }
24.
25. }
```

applicationContext.java

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:aop="http://www.springframework.org/schema/aop"
5.     xsi:schemaLocation="
6.         http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
7.         http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd">
8.     <!-- Beans -->
9.     <bean id="accBean" class="com.durgasoft.beans.Account">
10.        <property name="accNo" value="abc123"/>
11.        <property name="accName" value="Durga"/>
12.        <property name="accType" value="Savings"/>
13.        <property name="balance" value="20000"/>
14.    </bean>
15.    <bean id="chequeBean" class="com.durgasoft.beans.Cheque">
16.        <property name="cheque_No" value="xyz123"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
17.      <property name="amount" value="10000"/>
18.    </bean>
19.
20.    <!-- Target -->
21.    <bean id="target" class="com.durgasoft.bo.TransactionServiceImpl"/>
22.
23.    <!-- Advice -->
24.    <bean id="advice" class="com.durgasoft.advice.ChequeClearenceAdvice"/>
25.
26.    <!-- Proxy -->
27.    <bean id="proxy" class="org.springframework.aop.framework.ProxyFactoryBean">
28.        <property name="target" ref="target"/>
29.        <property name="interceptorNames">
30.            <list>
31.                <value>advice</value>
32.            </list>
33.        </property>
34.    </bean>
35.</beans>
```

Test.java

```
1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5.
6. import com.durgasoft.beans.Account;
7. import com.durgasoft.beans.Cheque;
8. import com.durgasoft.bo.TransactionService;
9.
10. public class Test {
11.
12.     public static void main(String[] args) {
13.         ApplicationContext context = new ClassPathXmlApplicationContext("applicationConte
xt.xml");
14.         Account account = (Account) context.getBean("accBean");
15.         Cheque cheque = (Cheque) context.getBean("chequeBean");
16.         TransactionService tx_Service = (TransactionService) context.getBean("proxy");
17.         tx_Service.debit(account, cheque);
18.     }
19.
20. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

AspectJ

AspectJ is well known in AOP language, it provides specialized syntax to express concerns. It also provides tools to add a concern into the system and enables crosscutting concern and modularization, such as logging, error checking and handling, and so on.

Spring is supporting AspectJ in the following two ways.

1. Declarative approach
2. @AspectJ annotation style approach

1. Declarative approach:

IN declarative approach, we will use Aspectj Namespace tags inorder to declare aspects, advices, Pointcuts,.....

In declarative configuration approach, all the aspect declarations are placed under the `<aop:config>` tag.

To use AOP namespace tags, we need to import the spring-aop schema.

applicationContext.xml

1. `<?xml version="1.0" encoding="UTF-8"?>`
2. `<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:aop="http://www.springframework.org/schema/aop" xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.1.xsd">`
3. `<aop:config>`
4. `<!-- contains aspect configuration and all method related configuration -->`
5. `</aop:config>`
6. `</beans>`

Note:The `aop:config` will contain all aspect configurations and all specific method-related configurations, such as around, pointcut, and so on.

1. Declaring Aspects:

To declare aspects by using AspectJ namespace tags we have to use the following tags.

`<aop:config>`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<aop:aspect id="--" ref="--">
```

```
</aop:aspect>
```

```
<aop:config>
```

- ✓ Where "id" attribute will take Aspect id value.

- ✓ Where "ref" attribute will take identity of the class which has declared in the configuration file by using <bean> tag in out side of <aop:aspect> tag.



EX:

```
1. <beans ..... >
2.   <aop:config>
3.     <aop:aspect id="loggingAspect" ref="loggingAspectBean">
4.     ...
5.   </aop:aspect>
6. </aop:config>
7.
8.   <bean id="loggingAspectBean" class="com.durgasoft.aspect.EmployeeCRUDLoggingAspect" />
9. </beans>
```

2. Declaring Pointcuts:

A pointcut helps in determining the join points to be executed with different advices.

To declare pointcuts we have to use the following tags in configuration file.

```
<aop:config>
  <aop:aspect id="----" ref="----">
    <aop:pointcut id="----" expression="----"/>
    ...
  </aop:aspect>
</aop:config>
```

- ✓ Where "id" attribute in <aop:pointcut> tag will take identity to the Pointcut.

- ✓ Where "expression" attribute in <aop:pointcut> will take AspectJ expression to define expression for the business methods which are required Advices.

- ✓ Where "expression" attribute will take "execution" function with an expression.

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:

```
<aop:pointcut id="businessService" expression="execution(* com.durgasoft.service.*.*(..))"/>
```

In the above code, expression will represent all java methods with any type of return type.

EX:

```
1. <aop:config>
2.   <aop:aspect id="loggingAspect" ref="loggingAspectBean">
3.
4.     <aop:pointcut id="loggingOperation" expression="execution(* com.durgasoft.service
   .EmployeeService.*(..))" />
5.
6.   </aop:aspect>
7. </aop:config>
8.
9. <bean id="loggingAspectBean" class="com.durgasoft.aspect.EmployeeCRUDLoggingAspe
   ct" />
```

Examples on Pointcut Expressions:

1. execution (* com.durgasoft.service.EmployeeService.*(..))

- The above Expression matches all of the methods declared in the EmployeeService interface
- The above expression matches methods with any modifier (public, protected, and private) and any return type.
- The two dots in the argument list match any number of arguments.

2.execution(* EmployeeService.*(..))

- The above Expression matches all methods of EmployeeService interface which are existed in the present package with any type of access modifier and with any return type.

3.execution(public * EmployeeService.*(..))

- The above Expression matches all public methods of EmployeeService interface with any return type.

4.execution(public Employee EmployeeService.*(..))

- The above Expression matches all public methods of EmployeeService interface with Employee return type.

5.execution(public Employee EmployeeService.*(Employee, ..))

- The above Expression matches all methods of EmployeeService interface with Employee return type and first parameter as Employee.

6.execution(public Employee EmployeeService.*(Employee, Integer))

- The above Expression matches all public methods of EmployeeService with Employee return type and with Employee as First parameter and Integer type parameter as second.
- Declaring Advices:

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Declaring Advices:

Spring AspectJ is supporting the following five advices .

1. <aop:before> It is applied before calling the actual business logic method.
2. <aop:after> It is applied after calling the actual business logic method.
3. <aop:after-returning> it is applied after calling the actual business logic method. It can be used to intercept the return value in advice.
4. <aop:around> It is applied before and after calling the actual business logic method.
5. <aop:after-throwing> It is applied if actual business logic method throws exception.

All the above advices tags contains "method" and "pointcut-ref" attributes, where "method" attribute will take advice method and "pointcut-ref" attribute will take Pointcut reference which we declared in Configuration file.

EX:

```

1. <beans xmlns="http://www.springframework.org/schema/beans"
2.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.   xmlns:aop="http://www.springframework.org/schema/aop"
4.   xsi:schemaLocation="http://www.springframework.org/schema/beans
5.   http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
6.   http://www.springframework.org/schema/aop/
7.   http://www.springframework.org/schema/aop/spring-aop-3.0.xsd ">
8.
9. <aop:config>
10.
11.   <!-- Spring AOP Pointcut definitions -->
12.   <aop:pointcut id="loggingOperation"
13.     expression="execution(* com.durgasoft.service.EmployeeService.*(..))" />
14.
15.   <!-- Spring AOP aspect -->
16.   <aop:aspect id="loggingAspect" ref="loggingAspectBean">
17.
18.     <!-- Spring AOP advises -->
19.     <aop:before pointcut-ref="loggingOperation" method="logBefore" />
20.     <aop:after pointcut-ref="loggingOperation" method="logAfter" />
21.
22.   </aop:aspect>
23.
24.
25. </aop:config>
26.
27. <!-- Spring AOP aspect instances -->
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

28. <bean id="loggingAspectBean" class="com.durgasoft.aspect.EmployeeCRUDLoggingAspect" />
29.
30. <!-- Target Object -->
31. <bean id="employeeManager" class="com.durgasoft.service.EmployeeServiceImpl" />
32.
33.</beans>
```

Steps to prepare Application by using AspectJ namespace tags

1. Declare Beans.
2. Declare Service interface
3. Declare Service interface implementation class.
4. Create Aspect class
5. Prepare Spring Configuration file.
6. Prepare Test Application

Example On AOP namespace tags

Employee.java

```

1. package com.durgasoft.beans;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.     public void setEno(int eno) {
13.         this.eno = eno;
14.     }
15.     public String getEname() {
16.         return ename;
17.     }
18.     public void setEname(String ename) {
19.         this.ename = ename;
20.     }
21.     public float getEsal() {
22.         return esal;
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

23. }
24. public void setEsal(float esal) {
25.     this.esal = esal;
26. }
27. public String getEaddr() {
28.     return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.     this.eaddr = eaddr;
32. }
33.
34.
35.}
```

EmployeeService.java



```

1. package com.durgasoft.service;
2.
3. import com.durgasoft.beans.Employee;
4.
5. public interface EmployeeService {
6.     public String createEmployee(Employee emp)throws Exception;
7.     public Employee searchEmployee(int eno);
8.     public String updateEmployee(Employee emp);
9.     public String deleteEmployee(Employee emp);
10.}
```

EmployeeServiceImpl.java



```

1. package com.durgasoft.service;
2.
3. import com.durgasoft.beans.Employee;
4.
5. public class EmployeeServiceImpl implements EmployeeService {
6.
7.     @Override
8.     public String createEmployee(Employee emp){
9.         System.out.println("Employee "+emp.getEno()+" Inserted Successfully from createEmployee()");
10.
11.        return "Success";
12.    }
13.}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

14. @Override
15. public Employee searchEmployee(int eno) {
16.     System.out.println("Employee "+eno+" Existed from serachEmployee()");
17.     return null;
18. }
19.
20. @Override
21. public String updateEmployee(Employee emp) {
22.     System.out.println("Employee "+emp.getEno()+" Updated Successfully from updateE
mployee()");
23.     return "Success";
24. }
25.
26. @Override
27. public String deleteEmployee(Employee emp) {
28.     System.out.println("Employee "+emp.getEno()+" Deleted Successfully from deleteEmp
loyee()");
29.     return null;
30. }
31.
32.}
```

LoggingAspectBean.java

```

1. package com.durgasoft.aspects;
2.
3. import org.aspectj.lang.JoinPoint;
4. import org.aspectj.lang.ProceedingJoinPoint;
5. import org.aspectj.lang.annotation.After;
6. import org.aspectj.lang.annotation.AfterReturning;
7. import org.aspectj.lang.annotation.AfterThrowing;
8. import org.aspectj.lang.annotation.Around;
9. import org.aspectj.lang.annotation.Aspect;
10. import org.aspectj.lang.annotation.Before;
11. public class LoggingAspectBean {
12.     public void before(JoinPoint jp) {
13.         System.out.println("Before "+jp.getSignature().getName()+" method execution");
14.     }
15.     public void after(JoinPoint jp) {
16.         System.out.println("After "+jp.getSignature().getName()+" method execution");
17.     }
18.     public void afterReturning(JoinPoint jp, Object result) {
19.         System.out.println("After Returning "+result+" from "+jp.getSignature().getName());
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

20.    }
21.   public void around(ProceedingJoinPoint jp) {
22.     System.out.println("Before "+jp.getSignature().getName()+"execution from around Adv
ice");
23.     try {
24.       jp.proceed();
25.     } catch (Throwable e) {
26.       e.printStackTrace();
27.     }
28.     System.out.println("After "+jp.getSignature().getName()+"execution from around Advis
e");
29.   }
30.   public void afterThrowing(JoinPoint jp, Throwable exception) {
31.     System.out.println("After throwing "+exception+" from "+jp.getSignature().getName()+""
method");
32.
33.   }
34. }
```

applicationContext.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <beans xmlns="http://www.springframework.org/schema/beans"
3.    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.    xmlns:aop="http://www.springframework.org/schema/aop"
5.    xsi:schemaLocation="
6.      http://www.springframework.org/schema/beans http://www.springframework.org/sche
ma/beans/spring-beans.xsd
7.      http://www.springframework.org/schema/aop http://www.springframework.org/schema/
aop/spring-aop.xsd">
8.
9.  <!-- beans -->
10. <bean id="empBean" class="com.durgasoft.beans.Employee">
11.   <property name="eno" value="111"/>
12.   <property name="ename" value="AAA"/>
13.   <property name="esal" value="5000"/>
14.   <property name="eaddr" value="Hyd"/>
15. </bean>
16. <!-- target Bean-->
17. <bean id="empService" class="com.durgasoft.service.EmployeeServiceImpl"/>
18.
19. <!-- Aspect bean -->
20. <bean id="loggingAspectBean" class="com.durgasoft.aspects.LoggingAspectBean"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
21.  
22.<aop:config>  
23. <aop:aspect id="loggingAspect" ref="loggingAspectBean">  
24.   <aop:pointcut expression="execution(* com.durgasoft.service.EmployeeService.*(..))  
   " id="empPointcut"/>  
25.  
26.   <aop:before method="before" pointcut-ref="empPointcut"/>  
27.   <aop:after method="after" pointcut-ref="empPointcut"/>  
28.   <aop:after-returning method="afterReturning" pointcut-  
     ref="empPointcut" returning="result"/>  
29.   <aop:around method="around" pointcut-ref="empPointcut"/>  
30.  
31.   <aop:after-throwing method="afterThrowing" throwing="exception" pointcut-  
     ref="empPointcut"/>  
32. </aop:aspect>  
33.</aop:config>  
34.  
35.</beans>
```

Test.java

```
1. package com.durgasoft.test;  
2.  
3. import org.springframework.context.ApplicationContext;  
4. import org.springframework.context.support.ClassPathXmlApplicationContext;  
5.  
6. import com.durgasoft.beans.Employee;  
7. import com.durgasoft.service.EmployeeService;  
8.  
9. public class Test {  
10.   public static void main(String[] args) {  
11.     ApplicationContext context = new ClassPathXmlApplicationContext("applicationConte  
xt.xml");  
12.     EmployeeService empService = (EmployeeService)context.getBean("empService");  
13.     Employee emp = (Employee) context.getBean("empBean");  
14.     String message = "";  
15.     try {  
16.       message = empService.createEmployee(emp);  
17.     } catch (Exception e) {  
18.     }  
19.   }  
20.   System.out.println(message);  
21. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

22.}

Example:**Show.java**

```
1. package com.durgasoft.beans;
2.
3. public class Show {
4.     private String name;
5.     private String time;
6.     private int price;
7.
8.     public String getName() {
9.         return name;
10.    }
11.    public void setName(String name) {
12.        this.name = name;
13.    }
14.    public String getTime() {
15.        return time;
16.    }
17.    public void setTime(String time) {
18.        this.time = time;
19.    }
20.    public int getPrice() {
21.        return price;
22.    }
23.    public void setPrice(int price) {
24.        this.price = price;
25.    }
26.}
```

**ShowService.java**

```
1. package com.durgasoft.service;
2.
3. import com.durgasoft.beans.Show;
4.
5. public interface ShowService {
6.     public String runShow(Show show) throws RuntimeException;
7. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

8.
9. ShowServiceImpl.java
10.-----
11. package com.durgasoft.service;
12.
13. import com.durgasoft.beans.Show;
14.
15. public class ShowServiceImpl implements ShowService {
16.
17.     @Override
18.     public String runShow(Show show) throws RuntimeException {
19.         System.out.println("*****Show "+show.getName()+" Start****");
20.         System.out.println("Show "+show.getName()+" is Running Successfully");
21.         if(!show.getName().equalsIgnoreCase("Mimicry")) {
22.             throw new RuntimeException();
23.         }
24.         System.out.println("*****Show "+show.getName()+" End****");
25.         return "success";
26.     }
27. }
```

ShowAspect.java



```

1. package com.durgasoft.aspect;
2.
3. import org.aspectj.lang.ProceedingJoinPoint;
4.
5. public class ShowAspect {
6.     public void before() {
7.         System.out.println("Get Tickets for the Show");
8.     }
9.     public void around(ProceedingJoinPoint jp) {
10.        System.out.println("Show is Ready To start, Take Chairs and Keep mobiles in Silent mode");
11.        try {
12.            jp.proceed();
13.        } catch (Throwable e) {
14.            e.printStackTrace();
15.        }
16.        System.out.println("Show Completed just now, Check your Laguages");
17.    }
18.    public void after() {
19.        System.out.println("Show is over , quit from Hall");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

20.    }
21.    public void afterReturning() {
22.        System.out.println("Tankq for attending Show");
23.    }
24.
25.    public void afterThrowing() {
26.        System.out.println("There is an Interruption in show, because, Show is not Mimicry sh
27.        ow");
28.    }

```

applicationContext.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <beans xmlns="http://www.springframework.org/schema/beans"
3.      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.      xmlns:aop="http://www.springframework.org/schema/aop"
5.      xsi:schemaLocation="
6.          http://www.springframework.org/schema/beans http://www.springframework.org/sche
ma/beans/spring-beans.xsd
7.          http://www.springframework.org/schema/aop http://www.springframework.org/schema/
aop/spring-aop.xsd">
8.      <!-- beans -->
9.      <bean id="showBean" class="com.durgasoft.beans.Show">
10.         <property name="name" value="Singing"/>
11.         <property name="time" value="7:30PM"/>
12.         <property name="price" value="1000"/>
13.     </bean>
14.
15.
16.     <!-- Target -->
17.     <bean id="showService" class="com.durgasoft.service.ShowServiceImpl"/>
18.
19.     <!-- aspect -->
20.     <bean id="showAspect" class="com.durgasoft.aspect.ShowAspect"/>
21.
22.     <aop:config>
23.         <aop:aspect id="mimicryShowAspect" ref="showAspect">
24.             <aop:pointcut expression="execution(public String com.durgasoft.service.ShowSe
rvice.runShow(com.durgasoft.beans.Show))" id="showPointcut"/>
25.
26.             <aop:before method="before" pointcut-ref="showPointcut"/>
27.             <!-- <aop:around method="around" pointcut-ref="showPointcut"/> -->

```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

28.      <aop:after method="after" pointcut-ref="showPointcut"/>
29.      <aop:after-returning method="afterReturning" pointcut-ref="showPointcut"/>
30.      <aop:after-throwing method="afterThrowing" pointcut-ref="showPointcut" />
31.    </aop:aspect>
32.  </aop:config>
33.</beans>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5.
6. import com.durgasoft.beans.Show;
7. import com.durgasoft.service.ShowService;
8.
9. public class Test {
10.
11.     public static void main(String[] args) {
12.         ApplicationContext context = new ClassPathXmlApplicationContext("applicationConte
xt.xml");
13.         Show show = (Show) context.getBean("showBean");
14.         ShowService showService = (ShowService) context.getBean("showService");
15.         try {
16.             showService.runShow(show);
17.         } catch (RuntimeException e) {
18.             //System.out.println(e.getMessage());
19.         }
20.     }
21. }
```



2. @AspectJ annotation style approach :

Spring Framework supporting Annotations to support AspectJ implementation in the form of "org.aspectj.lang.annotation" package.

Spring AspectJ AOP implementation provides the following annotations:

1. @Aspect: It will declare a class as an aspect.
2. @Pointcut: It will declare a pointcut expression.
3. @Before: It will declare before advice, It will be executed before executing the actual Business method.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4. @After: It will declare after advice, It will be executed after executing the actual Business method and before returning result.
5. @AfterReturning: It declares after returning advice, It will be executed after calling the actual Business method and after returning result.
6. @Around: It declares around advice, It will be executed before and after calling the actual Business method.
7. @AfterThrowing: It declares the throws advice, It will be executed if the actual Business method throws exception.

Note: To activate all the above annotations in Spring applications we have to use `<aop:aspectj-autoproxy>` tag in spring configuration file.

Example:

Account.java

```

1. package com.durgasoft.beans;
2.
3. public class Account {
4.     private String accNo;
5.     private String accName;
6.     private String accType;
7.     private int balance;
8.
9.     public String getAccNo() {
10.         return accNo;
11.     }
12.     public void setAccNo(String accNo) {
13.         this.accNo = accNo;
14.     }
15.     public String getAccName() {
16.         return accName;
17.     }
18.     public void setAccName(String accName) {
19.         this.accName = accName;
20.     }
21.     public String getAccType() {
22.         return accType;
23.     }
24.     public void setAccType(String accType) {
25.         this.accType = accType;
26.     }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

27. public int getBalance() {
28.     return balance;
29. }
30. public void setBalance(int balance) {
31.     this.balance = balance;
32. }
33.
34.
35.}
```

TransactionService.java

```

1. package com.durgasoft.service;
2.
3. import com.durgasoft.beans.Account;
4. import com.durgasoft.exceptions.InsufficientFundsException;
5.
6. public interface TransactionService {
7.     public String withdraw(Account acc, int wd_Amt) throws InsufficientFundsException;
8. }
```



TransactionServiceImpl.java

```

1. package com.durgasoft.service;
2.
3. import org.springframework.stereotype.Component;
4.
5. import com.durgasoft.beans.Account;
6. import com.durgasoft.exceptions.InsufficientFundsException;
7.
8. @Component("transaction")
9. public class TransactionServiceImpl implements TransactionService {
10.
11.     @Override
12.     public String withdraw(Account acc, int wd_Amt) throws InsufficientFundsException {
13.         String status = "";
14.         if(acc.getBalance() > wd_Amt) {
15.             int total_Bal = acc.getBalance() - wd_Amt;
16.             acc.setBalance(total_Bal);
17.             System.out.println("From withdraw(): Transaction Withdraw Completed ");
18.             status = "SUCCESS";
19.         }else {
20.             status = "FAILURE";
21.         }
22.     }
23. }
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

21.      throw new InsufficientFundsException("Funds are not Sufficient in Account");
22.  }
23.  return status;
24. }
25.

```

InsufficientFundsException.java

```

1. package com.durgasoft.exceptions;
2.
3. public class InsufficientFundsException extends Exception {
4.     public InsufficientFundsException(String desc) {
5.         super(desc);
6.     }
7. }

```

TransactionAspect.java

```

1. package com.durgasoft.aspect;
2.
3. import org.aspectj.lang.JoinPoint;
4. import org.aspectj.lang.ProceedingJoinPoint;
5. import org.aspectj.lang.annotation.After;
6. import org.aspectj.lang.annotation.AfterReturning;
7. import org.aspectj.lang.annotation.AfterThrowing;
8. import org.aspectj.lang.annotation.Around;
9. import org.aspectj.lang.annotation.Aspect;
10. import org.aspectj.lang.annotation.Before;
11. import org.springframework.stereotype.Component;
12.
13. import com.durgasoft.beans.Account;
14. import com.durgasoft.exceptions.InsufficientFundsException;
15. @Component("aspect")
16. @Aspect
17. public class TransactionAspect {
18.     @Before("execution(* com.durgasoft.service.TransactionService.*(..))")
19.     public void before(JoinPoint jp) {
20.         Object[] args = jp.getArgs();
21.         Account acc = (Account) args[0];
22.         System.out.println("Before Advice : Initial Balance :" + acc.getBalance());
23.     }
24.
25.     @After("execution(* com.durgasoft.service.TransactionService.*(..))")

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

26. public void after(JoinPoint jp) {
27.     Object[] args = jp.getArgs();
28.     Account acc = (Account) args[0];
29.     System.out.println("After Advice : Total Balance :" + acc.getBalance());
30. }
31.
32. @AfterReturning(pointcut="execution(* com.durgasoft.service.TransactionService.*(..))",
33.                 returning="result")
34. public void afterReturning(JoinPoint jp, String result) {
35.     System.out.println("After Returning Advice: Transaction Status :" + result);
36. }
37.
38. @Around("execution(* com.durgasoft.service.TransactionService.*(..))")
39. public void around(ProceedingJoinPoint jp) {
40.     System.out.println("Around Advice : Before " + jp.getSignature().getName() + " Method Execution");
41.     String status = "";
42.     try {
43.         status = (String) jp.proceed();
44.     } catch (Throwable e) {
45.         e.printStackTrace();
46.     }
47.     System.out.println("Around Advice : After " + jp.getSignature().getName() + " Method Execution");
48.     System.out.println("Around Advice : Transaction Status : " + status);
49. }
50.
51. //@AfterThrowing(pointcut="execution(* com.durgasoft.service.TransactionService.*(..))",
52. //                  throwing="exception")
53. public void afterThrowing(JoinPoint jp, InsufficientFundsException exception) {
54.     System.out.println("After Throwing Advice : " + exception.getClass().getName() + " In Transaction :" + exception.getMessage());
55. }
56.

```

applicationContext.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!--
3. <beans xmlns="http://www.springframework.org/schema/beans"
4.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5.   xmlns:aop="http://www.springframework.org/schema/aop"
6.   xsi:schemaLocation="
7.       http://www.springframework.org/schema/beans http://www.springframework.org/sche
ma/beans/spring-beans.xsd
8.       http://www.springframework.org/schema/aop http://www.springframework.org/schema/
aop/spring-aop.xsd">
9. -->
10.
11. <beans xmlns="http://www.springframework.org/schema/beans"
12.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
13.   xmlns:aop="http://www.springframework.org/schema/aop"
14.   xmlns:context="http://www.springframework.org/schema/context"
15.   xsi:schemaLocation="
16.       http://www.springframework.org/schema/beans http://www.springframework.org/sche
ma/beans/spring-beans.xsd
17.       http://www.springframework.org/schema/aop http://www.springframework.org/schema/
aop/spring-aop.xsd
18. http://www.springframework.org/schema/context
19.   http://www.springframework.org/schema/context/spring-context.xsd">
20.
21. <context:annotation-config/>
22. <context:component-scan base-package="com.durgasoft.service"/>
23. <context:component-scan base-package="com.durgasoft.aspect"/>
24. <aop:aspectj-autoproxy/>
25. <!-- beans -->
26. <bean id="accBean" class="com.durgasoft.beans.Account">
27.   <property name="accNo" value="abc123"/>
28.   <property name="accName" value="Durga"/>
29.   <property name="accType" value="Savings"/>
30.   <property name="balance" value="20000"/>
31. </bean>
32. <!--
33.
34. <bean id="transaction" class="com.durgasoft.service.TransactionServiceImpl"/>
35.
36.
37. <bean id="txAspect" class="com.durgasoft.aspect.TransactionAspect"/>
38. -->
39.</beans>
```

Test.java

```
1. package com.durgasoft.test;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.  
3. import org.springframework.context.ApplicationContext;  
4. import org.springframework.context.support.ClassPathXmlApplicationContext;  
5.  
6. import com.durgasoft.beans.Account;  
7. import com.durgasoft.exceptions.InsufficientFundsException;  
8. import com.durgasoft.service.TransactionService;  
9.  
10. public class Test {  
11.  
12.     public static void main(String[] args) {  
13.         ApplicationContext context = new ClassPathXmlApplicationContext("applicationConte  
xt.xml");  
14.         Account acc = (Account) context.getBean("accBean");  
15.         TransactionService txService = (TransactionService) context.getBean("transaction");  
16.         try {  
17.             txService.withdraw(acc,50000);  
18.         } catch (InsufficientFundsException e) {  
19.             //e.printStackTrace();  
20.         }  
21.  
22.     }  
23.}
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGAGHOSH

{SPRING ORM}

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING ORM MODULE INDEX

- | | |
|--|----------|
| 1. Spring ORM..... | PAGE 363 |
| 2. Application on Spring-Hibernate integration..... | PAGE 371 |
| 3. JPA[Java Persistence API] | PAGE 377 |
| 4. Integration of JPA with Spring application in ORM Module..... | PAGE 391 |

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring ORM

Spring ORM:

In enterprise applications both the data models are having their own approaches to represent data in effective manner, these differences are able to provide Paradiagnm Mismatches, these mismatches are able to reduce data persistency in enterprise applications.

In general, Object Oriented Data Model and Relational data model are having the following mismatches.

1. Granularity Mismatch
2. Sub types mismatch
3. Associations Mismatch
4. Identity Mismatch

To improve Data persistency in Enterprise applications we have to resolve the above specified mismatches between Data models, for this, we have to use "ORM" implementations.

To implement ORM in Enterprise applications we have to use the following ORM implementations.

1. EJBs-Entity Beans
2. JPA
3. Hibernate
4. iBatis
5. JDO

If we want to use Hibernate in enterprise applications then we have to use the following steps.

- 1) Persistence Class or Object.
- 2) Prepare Mapping File.
- 3) Prepare Hibernate Configuration File
- 4) Prepare Hibernate Client Application

To prepare Hibernate Client Application we have to use the following steps.

1. Create Configuration class object
2. Create SessionFactory object
3. Create Session Object
4. Create Transaction object if it is required.
5. Persistence operations
6. Close Session Factory and Session objects.

Example:

1. Configuration `cfg = new Configuration();`
2. `cfg.configure("hibernate.cfg.xml");`
3. SessionFactory `session_Factory = cfg.buildSessionFactory();`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
4. Session session = session_Factory.openSession();  
5. Transaction tx = session.beginTransaction();  
6. Employee emp = new Employee();  
7. emp.setEno(111);  
8. emp.setEname("AAA");  
9. emp.setEsal(5000);  
10. emp.setEaddr("Hyd");  
11. session.save(emp);  
12. tx.commit();  
13. System.out.println("Employee Record inserted Successfully");  
14. session.close();  
15. session_Factory.close();
```

To remove the above boilerplate code, SPRING Framework has provided ORM Module. Spring has provided the complete ORM module in the form of "org.springframework.orm" package.

To abstract the above boilerplate code Spring-ORM module has provided a predefined class in the form of "org.springframework.orm.hibernate4.HibernateTemplate" w.r.t Hibernate4 version

Note: If we use Hibernate3.x version then we have to use "org.springframework.orm.hibernate3.HibernateTemplate" class.

org.springframework.orm.hibernate4.HibernateTemplate class has provided the following methods inorder to perform persistence operations.

1. public void persist(Object entity)
2. public Serializable save(Object entity)
3. public void saveOrUpdate(Object entity)
4. public void update(Object entity)
5. public void delete(Object entity)
6. public Object get(Class entityClass, Serializable id)
7. public Object load(Class entityClass, Serializable id)
8. public List loadAll(Class entityClass)

If we want to Integrate Hibernate with Spring then we have to use the following steps.

- 1) Create Java Project with both Spring[including ORM] and Hibernate Libraries.
- 2) Create Bean/POJO class.
- 3) Prepare Hibernate Mapping File
- 4) Create DAO interface with persistence methods.
- 5) Create DAO implementation class with HibernateTemplate as property.
- 6) Prepare Spring Configuration File
- 7) Prepare Client Application

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. Create Java Project with both Spring[including ORM] and Hibernate Libraries.

Prepare JAVA project in Eclipse IDE and add the following JAR files to Buildpath in the form of the following Libraries.

Spring4_Lib:

- + spring-aop-4.0.4.RELEASE.jar
- + spring-beans-4.0.4.RELEASE.jar
- + spring-context-4.0.4.RELEASE.jar
- + spring-context-support-4.0.4.RELEASE.jar
- + spring-core-4.0.4.RELEASE.jar
- + spring-expression-4.0.4.RELEASE.jar
- + spring-jdbc-4.0.4.RELEASE.jar
- + commons-io-2.6.jar
- + commons-logging-1.2.jar
- + spring-tx-4.0.4.RELEASE.jar
- + spring-aspects-4.0.4.RELEASE.jar
- + spring-orm-4.0.4.RELEASE.jar

Hibernate4_Lib:

- + ojdbc6.jar
- + antlr-2.7.7.jar
- + dom4j-1.6.1.jar
- + hibernate-commons-annotations-4.0.5.Final.jar
- + hibernate-core-4.3.11.Final.jar
- + hibernate-jpa-2.1-api-1.0.0.Final.jar
- + jandex-1.1.0.Final.jar
- + javassist-3.18.1-GA.jar
- + jboss-logging-3.1.3.GA.jar
- + jboss-logging-annotations-1.2.0.Beta1.jar
- + jboss-transaction-api_1.2_spec-1.0.0.Final.jar
- + hibernate-entitymanager-4.3.11.Final.jar

2. Create Bean/POJO class.

```
1. public class Student{  
2.     private String sid;  
3.     private String sname;  
4.     private String saddr;  
5.     setXXX() and getXXX()  
6. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. Prepare Hibernate Mapping File:

Student.hbm.xml

```

1. <!DOCTYPE ---- >
2. <hibernate-mapping>
3.   <class name="com.durgasoft.pojo.Student" table="student">
4.     <id name="sid" column="SID"/>
5.     <property name="sname" column="SNAME"/>
6.     <property name="saddr" column="SADDR"/>
7.   </class>
8. </hibernate-mapping>
```

4. Create DAO interface with persistence methods.

The main intention of DAO interface is to declare all Services.

EX:

```

1. public interface StudentDao {
2.   public String insert(Student std);
3.   public String update(Student std);
4.   public String delete(Student std);
5.   public Employee getStudent(int eno);
6. }
```

5. Create DAO implementation class with HibernateTemplate as property.

The main intention of DAO implementation class is to implement all DAO methods. In DAO implementation class every DAO method must be declared with `@Transactional` annotation inorder to activate Spring Transaction Service.

Note: If we use `@Transactional` annotation then it is not required to create Transaction object explicitly and it is not required to perform commit and rollback operations explicitly.

In DAO implementation class we must declare `HibernateTemplate` property and its respective `setXXX()` method inorder to inject `HibernateTemplate` object.

Example:

```

1. public class StudentDaolmpl implements StudentDao {
2.   String status = "";
3.   private HibernateTemplate hibernateTemplate;
4.   public void setHibernateTemplate(HibernateTemplate hibernateTemplate) {
5.     this.hibernateTemplate = hibernateTemplate;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6.    }
7.
8.    @Transactional
9.    public String insert(Student std) {
10.       try {
11.           hibernateTemplate.save(std);
12.           status = "Insertion Success";
13.       }catch(Exception ex) {
14.           ex.printStackTrace();
15.           status = "Insertion Failure";
16.       }
17.       return status;
18.   }
19.
20.   @Transactional
21.   public String update(Student std) {
22.       try {
23.           hibernateTemplate.update(std);
24.           status = "Updations Success";
25.       }catch(Exception ex) {
26.           ex.printStackTrace();
27.           status = "Updations Failure";
28.       }
29.       return status;
30.   }
31.
32.   @Transactional
33.   public String delete(Student std) {
34.       try {
35.           hibernateTemplate.delete(std);
36.           status = "Deletion Success";
37.       }catch(Exception ex) {
38.           ex.printStackTrace();
39.           status = "Deletion Failure";
40.       }
41.       return status;
42.
43.   }
44.
45.   @Transactional
46.   public Employee getStudent(int sid) {
47.       Student std = null;
48.       try {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

49.     std = (Student)hibernateTemplate.get(Student.class, sid);
50. }catch(Exception ex) {
51.     ex.printStackTrace();
52. }
53. return std;
54. }
55.

```

6. Prepare Spring Configuration File:

In Spring Configuration File we must configure the following beans

1. DriverManagerDataSource
2. LocalSessionFactoryBean
3. HibernateTransactionManager
4. HibernateTemplate
5. StudentDaoImpl

- ✓ Where org.springframework.jdbc.datasource.DriverManagerDataSource configuration is able to provide Spring inbuilt Connectionpooling Mechanism and it will include the properties like "driverClassName, url, username, password".
- ✓ Where org.springframework.orm.hibernate4.LocalSessionFactoryBean configuration is able to create SessionFactory object by including the properties like
 1. **dataSource** : represents DataSource bean which was configured in Spring Configuration file.
 2. **mappingResources**: It will take list of values contains mapping files in the form of "<list>" tag.
 3. **hibernateProperties**: It will take hibernate properties in the form of "<props>" tag, it must includes mainly "hibernate.dialect" property.
- ✓ Where org.springframework.orm.hibernate4.HibernateTransactionManager configuration will activate Transaction Manager inorder to provide Transaction Support and it will include "sessionFactory" property.
- ✓ Where org.springframework.orm.hibernate4.HibernateTemplate configuration will provide HibernateTemplate object inorder to perform persistence operations and it will include "sessionFactory" and "checkWriteOperations" with false value.
- ✓ Where com.durgasoft.dao.StudentDaoImpl configuration will provide DAO object inorder to access Dao methods and it will include "hibernateTemplate" property.

Note: To use @Transactional annotation in DAO methods , we must use "<tx:annotation-driven/>" tag in spring configuration file.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**applicationContext.xml**

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:aop="http://www.springframework.org/schema/aop"
5.   xmlns:tx="http://www.springframework.org/schema/tx"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/tx
10.    http://www.springframework.org/schema/tx/spring-tx.xsd
11.    http://www.springframework.org/schema/aop
12.    http://www.springframework.org/schema/aop/spring-aop.xsd">
13.
14. <bean name="dataSource" class="org.springframework.jdbc.datasource.DriverManagerD
ataSource">
15.   <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
16.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
17.   <property name="username" value="system"/>
18.   <property name="password" value="durga"/>
19. </bean>
20. <bean name="sessionFactory" class="org.springframework.orm.hibernate4.LocalSessionF
actoryBean">
21.   <property name="dataSource" ref="dataSource"/>
22.   <property name="mappingResources">
23.     <list>
24.       <value>Student.hbm.xml</value>
25.     </list>
26.   </property>
27.   <property name="hibernateProperties">
28.     <props>
29.       <prop key="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</prop>
30.       <prop key="hibernate.show_sql">true</prop>
31.     </props>
32.   </property>
33. </bean>
34. <tx:annotation-driven/>
35. <bean id="transactionManager" class="org.springframework.orm.hibernate4.HibernateTra
nsactionManager">
36.   <property name="sessionFactory" ref="sessionFactory"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

37. </bean>
38.
39. <bean name="hibernateTemplate" class="org.springframework.orm.hibernate4.HibernateTemplate">
40.   <property name="sessionFactory" ref="sessionFactory"/>
41.   <property name="checkWriteOperations" value="false"></property>
42. </bean>
43. <bean name="stdDao" class="com.durgasoft.dao.StudentDaoImpl">
44.   <property name="hibernateTemplate" ref="hibernateTemplate"/>
45. </bean>
46. </beans>
```

7. Prepare Client Application

The main intention of Client Application is to get Dao object and to access Dao object.

Example:

```

1. ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
2. StudentDao stdDao = (StudentDao)context.getBean("stdDao");
3. Student std = new Student();
4. std.setSid("S-111");
5. std.setSname("AAA");
6. std.setSaddr("Hyd");
7. String status = stdDao.insert(std);
8. System.out.println(status);
9. or
10. Student std = (Student)stdDao.getStudent(Student.class,"S-111");
11. System.out.println("Student Details");
12. System.out.println("-----");
13. System.out.println("Student Id : "+std.getSid());
14. System.out.println("Student Name : "+std.getSname());
15. System.out.println("Student Address : "+std.getSaddr());
16. or
17. Student std = new Student();
18. std.setSid("S-111");
19. std.setSname("BBB");
20. std.setSaddr("Hyd");
21. String status = stdDao.update(std);
22. System.out.println(status);
23. or
24. String status = stdDao.delete("S-111");
25. System.out.println(status);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Application on Spring-Hibernate integration

Application on Spring-Hibernate integration:

Employee.java



```
1. package com.durgasoft.pojo;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.     public void setEno(int eno) {
13.         this.eno = eno;
14.     }
15.     public String getEname() {
16.         return ename;
17.     }
18.     public void setEname(String ename) {
19.         this.ename = ename;
20.     }
21.     public float getEsal() {
22.         return esal;
23.     }
24.     public void setEsal(float esal) {
25.         this.esal = esal;
26.     }
27.     public String getEaddr() {
28.         return eaddr;
29.     }
30.     public void setEaddr(String eaddr) {
31.         this.eaddr = eaddr;
32.     }
33.
34.     public String toString() {
35.         return "["+eno+","+ename+","+esal+","+eaddr+"]";
36.     }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

37.
38.}**EmployeeDao.java**

```
1. package com.durgasoft.dao;
2.
3. import com.durgasoft.pojo.Employee;
4.
5. public interface EmployeeDao {
6.     public String insert(Employee e);
7.     public String update(Employee e);
8.     public String delete(Employee e);
9.     public Employee getEmployee(int eno);
10.}
```

EmployeeDaoImpl.java

```
1. package com.durgasoft.dao;
2. import org.hibernate.FlushMode;
3. import org.hibernate.Transaction;
4. import org.springframework.orm.hibernate4.HibernateTemplate;
5. import org.springframework.transaction.annotation.Transactional;
6.
7. import com.durgasoft.pojo.Employee;
8.
9. public class EmployeeDaoImpl implements EmployeeDao {
10.     String status = "";
11.     private HibernateTemplate hibernateTemplate;
12.     public void setHibernateTemplate(HibernateTemplate hibernateTemplate) {
13.         this.hibernateTemplate = hibernateTemplate;
14.
15.     }
16.
17.     @Override
18.     @Transactional
19.     public String insert(Employee e) {
20.         try {
21.             hibernateTemplate.save(e);
22.             status = "Insertion Success";
23.         }catch(Exception ex) {
24.             ex.printStackTrace();
25.             status = "Insertion Failure";
26.         }
27.     }
28. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
26.    }
27.    return status;
28. }
29.
30. @Override
31. @Transactional
32. public String update(Employee e) {
33.     try {
34.         hibernateTemplate.update(e);
35.         status = "Updations Success";
36.     }catch(Exception ex) {
37.         ex.printStackTrace();
38.         status = "Updations Failure";
39.     }
40.     return status;
41. }
42.
43. @Override
44. @Transactional
45. public String delete(Employee e) {
46.     try {
47.         hibernateTemplate.delete(e);
48.         status = "Deletion Success";
49.     }catch(Exception ex) {
50.         ex.printStackTrace();
51.         status = "Deletion Failure";
52.     }
53.     return status;
54.
55. }
56.
57. @Override
58. @Transactional
59. public Employee getEmployee(int eno) {
60.     Employee emp = null;
61.     try {
62.         emp = (Employee)hibernateTemplate.get(Employee.class, eno);
63.     }catch(Exception ex) {
64.         ex.printStackTrace();
65.     }
66.     return emp;
67. }
68. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Employee.hbm.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Employee" table="emp1">
7.     <id name="eno" column="ENO"/>
8.     <property name="ename" column="ENAME"/>
9.     <property name="esal" column="ESAL"/>
10.    <property name="eaddr" column="EADDR"/>
11.  </class>
12. </hibernate-mapping>
```

applicationContext.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:aop="http://www.springframework.org/schema/aop"
5.   xmlns:tx="http://www.springframework.org/schema/tx"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/tx
10.    http://www.springframework.org/schema/tx/spring-tx.xsd
11.    http://www.springframework.org/schema/aop
12.    http://www.springframework.org/schema/aop/spring-aop.xsd">
13.
14.
15. <bean name="dataSource" class="org.springframework.jdbc.datasource.DriverManagerD
ataSource">
16.   <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
17.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
18.   <property name="username" value="system"/>
19.   <property name="password" value="durga"/>
20. </bean>
21. <bean name="sessionFactory" class="org.springframework.orm.hibernate4.LocalSessionF
actoryBean">
22.   <property name="dataSource" ref="dataSource"/>
23.   <property name="mappingResources">
24.     <list>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

25.    <value>Employee.hbm.xml</value>
26.  </list>
27. </property>
28. <property name="hibernateProperties">
29.   <props>
30.     <prop key="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</prop>
31.     <!-- <prop key="hibernate.current_session_context_class">thread</prop> -->
32.     <prop key="hibernate.show_sql">true</prop>
33.   </props>
34. </property>
35. </bean>
36. <tx:annotation-driven/>
37. <bean id="transactionManager" class="org.springframework.orm.hibernate4.HibernateTransactionManager">
38.   <property name="sessionFactory" ref="sessionFactory"/>
39. </bean>
40.
41. <bean name="hibernateTemplate" class="org.springframework.orm.hibernate4.HibernateTemplate">
42.   <property name="sessionFactory" ref="sessionFactory"/>
43.   <property name="checkWriteOperations" value="false"></property>
44. </bean>
45. <bean name="empDao" class="com.durgasoft.dao.EmployeeDaoImpl">
46.   <property name="hibernateTemplate" ref="hibernateTemplate"/>
47. </bean>
48. </beans>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5. import org.springframework.orm.hibernate4.HibernateTemplate;
6.
7. import com.durgasoft.dao.EmployeeDao;
8. import com.durgasoft.pojo.Employee;
9.
10. public class Test {
11.
12.   public static void main(String[] args) throws Exception {
13.     ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14. EmployeeDao empDao = (EmployeeDao)context.getBean("empDao");
15. Employee emp = new Employee();
16. emp.setEno(111);
17. emp.setEname("AAA");
18. emp.setEsal(5000);
19. emp.setEaddr("Hyd");
20. String status = empDao.insert(emp);
21. System.out.println(status);
22. System.out.println(empDao.getEmployee(111));
23.
24.
25. }
26.
27.}
```

DURGASURY

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

JPA[Java Persistence API]

JPA[Java Persistence API]

Introduction:

- JPA is a an API , it can be used to perform database operations in enterprise applications with ORM implementation tools.
- JPA was provided by J2EE along with EJB3.0 version as a persistence mechanism.
- JPA is a specification provided by SUN Microsystems and it is implemented by some third party vendors like JBOSS, Eclipse Foundations, Apache.....
- JPA is following ORM rules regulations to achieve data persistency in enterprise applications and it is implemented by the following tools.
 1. Hibernate -----> JBOSS
 2. EclipseLink -----> Eclipse Foundation
 3. Open JPA -----> Apache Software Foundations

Note: If we want to use JPA in enterprise applications then we must use either of the JPA implementations.

If we want to prepare JPA applications with "Hibernate" JPA provider then we have to use the following steps.

1. Create Java project in Eclipse with JPA library which includes all Hibernate JARs.
2. Create Entity class under src folder.
3. Create mapping File or Use JPA annotations in POJO class.
4. Create JPA configuration File[persistence.xml]
5. Create Test Application.

1. Create Java project in Eclipse with JPA library which includes all Hibernate JARs.

This step is same as Java project creation and it will include JPA provider Library that is Hibernate jars.

2. Create Entity class under src folder.

Student.java

```
1. package com.durgasoft.entity;  
2. public class Student{
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. private String sid;
4. private String sname;
5. private String saddr;
6. setXXX() and getXXX()
7. }
```

3. Create mapping File or Use JPA annotations in POJO class.

It is same as Hibernate mapping file, it will provide mapping between Object Oriented Data model elements like class, Id property, normal properties with the relational data model elements like Table name, Primary Key Columns, normal columns,....

EX:

Student.xml

```

1. <hibernate-mapping>
2.   <class name="com.durgasoft.entity.Student" table="student">
3.     <id name="sid" column="SID"/>
4.     <property name="sname" column="SNAME"/>
5.     <property name="saddr" column="SADDR"/>
6.   </class>
7. </hibernate-mapping>
```

4. Create JPA configuration File[persistence.xml]

JPA configuration File is same as Hibernate Configuration File, it include all JPA configuration details which are required to interact with database .

IN general, we will provide the following configuration details in JPA configuration file.

1. Jdbc Parameters like Driver class name, Driver URL, Database user name, Database password.
2. Dialect configurations
3. Mapping File or Annotated classes configuration
4. Cache Mechanisms configurations
5. Transactions configurations

The default name of the JPA configuration file is "persistence.xml".

Ex persistence.xml:

```
1. <persistence>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. <persistence-unit name="std">
3.   <!-- <class>com.durgasoft.entity.Student</class>-->
4.   <mapping-file>Student.xml</mapping-file>
5.   <properties>
6.     <property name="javax.persistence.jdbc.driver" value="oracle.jdbc.OracleDriver"/>
7.     <property name="javax.persistence.jdbc.url" value="jdbc:oracle:thin:@localhost:1521
:xe"/>
8.     <property name="javax.persistence.jdbc.user" value="system"/>
9.     <property name="javax.persistence.jdbc.password" value="durga"/>
10.    <property name="hibernate.dialect" value="org.hibernate.dialect.Oracle10gDialect"/>
11.    <property name="hibernate.show_sql" value="true"/>
12.    <property name="hibernate.format_sql" value="true"/>
13.  </properties>
14. </persistence-unit>
15. </persistence>
```

- ✓ Where <persistence> tag is root tag in JPA configuration File.
- ✓ Where <mapping-file> tag is able to take mapping file configuration
- ✓ where <propertis> tag will include JPA properties.
- ✓ Where <property> tag will take a single JPA property like driver class name, driver url,....

5. Create Test Application:

The main intention of Test /Client application is to perform persistence operations .

To prepare Test application in JPA we have to use the following steps.

1. Create EntityManagerFactory Object.
2. Create EntityManager Object.
3. Create EntityTransaction Object as per the requirement
4. Perform Persistence operation
5. Perform Commit or rollback operations if we use EntityTransaction.

1. Create EntityManagerFactory Object:

javax.persistence.EntityManagerFactory is a Factory class, it able to manage no of EntityManager object.

To get EntityManagerFactory class object we have to use the following method from javax.persistence.Persistence class.

```
public static EntityManagerFactory createEntityManagerFactory(String
persistence_Unit_Name);
```

EX: EntityManagerFactory factory = Persistence.createEntityManagerFactory("std");

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Create EntityManager Object:

`javax.persistence.EntityManager` is an interface, it able to provide predefined Library to perform persistence operations.

To get EntityManager object we have to use the following method from EntityManagerFactory.

```
public EntityManager createEntityManager()
```

EX: `EntityManager entManager = factory.createEntityManager();`

3. Create EntityTransaction Object as per the requirement

`javax.persistence.EntityTransaction` is a class, it able to provide Transaction support in JPA applications inorder to perform persistence operations.

To get EntityTransaction object we have to use the following method from EntityManager.

```
public EntityTransaction getTransaction()
```

EX: `EntityTransaction entTransaction = entManager.getTransaction();`

Note: EntityTransaction contains the following methods inorder to complete Transaction.

```
public void commit()  
public void rollback()
```

Note: EntityTransaction is required for only non select operations only, not for select operations.

4. Perform Persistence operation:

To perform Persistence operations we have to use the following methods from EntityManager object.

1. `public Object find(Class entity_Class, Serializable pk_Value)`
2. `public void persist(Object obj)`
3. `public void remove(Object obj)`

Note: To perform Updations , first we have to get Entity object from Database table by using `find()` method then we have to use set New data to Entity Object then perform commit operation.

EX:

1. `EntityTransaction entTransaction = entManager.getTransaction();`
2. `entTransaction.begin();`

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. Student std = (Student)entManager.find(Student.class, "S-111");
4. std.setSname("BBB");
5. std.setSaddr("Sec");
6. entTransaction.commit();
7. System.out.println("Student Updated Successfully");

```

EX:



Test.java

```

1. public class Test {
2.
3.     public static void main(String[] args) throws Exception {
4.         EntityManagerFactory factory = Persistence.createEntityManagerFactory("std");
5.         EntityManager entManager = factory.createEntityManager();
6.         Student std = new Student();
7.         std.setSid("S-111");
8.         std.setSname("AAA");
9.         std.setSaddr("Hyd");
10.        EntityTransaction tx = entManager.getTransaction();
11.        tx.begin();
12.        entManager.persist(std);
13.        tx.commit();
14.        System.out.println("Student Inserted Succssfully");
15.    }
16.}

```

Simple JPA Example with XML Mapping File:

Employee.java

```

1. package com.durgasoft.pojo;
2. public class Employee {
3.     private int eno;
4.     private String ename;
5.     private float esal;
6.     private String eaddr;
7.     public int getEno() {
8.         return eno;
9.     }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10. public void setEno(int eno) {
11.     this.eno = eno;
12. }
13. public String getEname() {
14.     return ename;
15. }
16. public void setEname(String ename) {
17.     this.ename = ename;
18. }
19. public float getEsal() {
20.     return esal;
21. }
22. public void setEsal(float esal) {
23.     this.esal = esal;
24. }
25. public String getEaddr() {
26.     return eaddr;
27. }
28. public void setEaddr(String eaddr) {
29.     this.eaddr = eaddr;
30. }
31.

```

Employee.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.pojo.Employee" table="emp1">
7.   <id name="eno" column="ENO"/>
8.   <property name="ename" column="ENAME"/>
9.   <property name="esal" column="ESAL"/>
10.  <property name="eaddr" column="EADDR"/>
11. </class>
12. </hibernate-mapping>

```

src/META-INF/persistence.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <persistence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3.   xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd"
4.   version="2.0" xmlns="http://java.sun.com/xml/ns/persistence">
5. <persistence-unit name="emp">
6.   <!-- <class>com.durgasoft.pojo.Employee</class>-->
7.   <mapping-file>Employee.xml</mapping-file>
8.   <properties>
9.     <property name="javax.persistence.jdbc.driver" value="oracle.jdbc.OracleDriver"/>
10.    <property name="javax.persistence.jdbc.url" value="jdbc:oracle:thin:@localhost:1521
:xe"/>
11.    <property name="javax.persistence.jdbc.user" value="system"/>
12.    <property name="javax.persistence.jdbc.password" value="durga"/>
13.    <property name="hibernate.dialect" value="org.hibernate.dialect.Oracle10gDialect"/>
14.    <property name="hibernate.show_sql" value="true"/>
15.    <property name="hibernate.format_sql" value="true"/>
16.  </properties>
17. </persistence-unit>
18. </persistence>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import javax.persistence.EntityManager;
4. import javax.persistence.EntityManagerFactory;
5. import javax.persistence.EntityTransaction;
6. import javax.persistence.Persistence;
7.
8. import com.durgasoft.pojo.Employee;
9.
10. public class Test {
11.
12.     public static void main(String[] args) throws Exception {
13.         EntityManagerFactory factory = Persistence.createEntityManagerFactory("emp");
14.         EntityManager entManager = factory.createEntityManager();
15.         Employee emp = new Employee();
16.         emp.setEno(111);
17.         emp.setEname("AAA");
18.         emp.setEsal(5000);
19.         emp.setEaddr("Hyd");
20.         EntityTransaction tx = entManager.getTransaction();
21.         tx.begin();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

22.     emManager.persist(emp);
23.     tx.commit();
24.     System.out.println("Employee Inserted Succssfully");
25. }
26.

```

Simple JPA Example with Annotations:

If we want to use Annotations in JPA application then we have to use the following steps.

1. Use javax.persistence provided annotations in Entity class:

1. @Entity
2. @Table
3. @Id
4. @Column

2. Configure Annotated class in persistence.xml file:

```

1. <persistence>
2.   <persistence-unit name="std">
3.     <class>com.durgasoft.entity.Student</class>
4.     -----
5.   </persistence-unit>
6. </persistence>

```

Example:

Employee.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp2")
10. public class Employee {
11.   @Id
12.   @Column(name="ENO")
13.   private int eno;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

14. @Column(name="ENAME")
15. private String ename;
16. @Column(name="ESAL")
17. private float esal;
18. @Column(name="EADDR")
19. private String eaddr;
20. public int getEno() {
21.     return eno;
22. }
23. public void setEno(int eno) {
24.     this.eno = eno;
25. }
26. public String getEname() {
27.     return ename;
28. }
29. public void setEname(String ename) {
30.     this.ename = ename;
31. }
32. public float getEsal() {
33.     return esal;
34. }
35. public void setEsal(float esal) {
36.     this.esal = esal;
37. }
38. public String getEaddr() {
39.     return eaddr;
40. }
41. public void setEaddr(String eaddr) {
42.     this.eaddr = eaddr;
43. }
44.
45.
46.
}

```

~~src/META-INF/persistence.xml~~

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <persistence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.   xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/p
ersistence/persistence_2_0.xsd"
4.   version="2.0" xmlns="http://java.sun.com/xml/ns/persistence">
5. <persistence-unit name="emp">
6.   <class>com.durgasoft.pojo.Employee</class>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

7.    <!-- <mapping-file>Employee.xml</mapping-file> -->
8.    <properties>
9.        <property name="javax.persistence.jdbc.driver" value="oracle.jdbc.OracleDriver"/>
10.       <property name="javax.persistence.jdbc.url" value="jdbc:oracle:thin:@localhost:1521
     :xe"/>
11.       <property name="javax.persistence.jdbc.user" value="system"/>
12.       <property name="javax.persistence.jdbc.password" value="durga"/>
13.       <property name="hibernate.dialect" value="org.hibernate.dialect.Oracle10gDialect"/>
14.       <property name="hibernate.show_sql" value="true"/>
15.       <property name="hibernate.format_sql" value="true"/>
16.   </properties>
17. </persistence-unit>
18.</persistence>
```

Test.java



```

1. package com.durgasoft.test;
2.
3. import javax.persistence.EntityManager;
4. import javax.persistence.EntityManagerFactory;
5. import javax.persistence.EntityTransaction;
6. import javax.persistence.Persistence;
7.
8. import com.durgasoft.pojo.Employee;
9.
10. public class Test {
11.
12.     public static void main(String[] args) throws Exception {
13.         EntityManagerFactory factory = Persistence.createEntityManagerFactory("emp");
14.         EntityManager entManager = factory.createEntityManager();
15.         Employee emp = new Employee();
16.         emp.setEno(111);
17.         emp.setEname("AAA");
18.         emp.setEsal(5000);
19.         emp.setEaddr("Hyd");
20.         EntityTransaction tx = entManager.getTransaction();
21.         tx.begin();
22.         entManager.persist(emp);
23.         tx.commit();
24.         System.out.println("Employee Inserted Succssfully");
25.     }
26. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

JPA With ECLIPSE Link Implementattion:

If we want to use JPA with EclipseLink implementation then we have to use the following steps.

1. Create JPA project.
2. Create Entity Class with Annotations
3. Create JPA configuration File
4. Create Test Application

1. Create JPA project.

1. Right Click on "Project Explorer".
2. Select on "New".
3. Select "Others"
4. Search and Select JPA Project
5. Click on "Next" button.
6. Provide package name:app6
7. Click on "next" button.
8. Click on "Next" button.
9. Click on "Download Libraries" icon.
10. Select EclipseLink2.5.2 library.
11. Click on "Next" button.
12. Select "Check box" of Accept Licence of this Agrement.
13. Click on "Finish" Button.
14. Click on "Finish" button.
15. Click on "Open Perspective".

With these steps JPA project will be created in projects Explorer part..

2. Create Entity Class with Annotations

Create package "com.durgasoft.entity" under src folder and create Entity class.

Employee.java

```
1. package com.durgasoft.entity;
2. import java.io.Serializable;
3. import java.lang.String;
4. import javax.persistence.*;
5. @Entity
6. @Table(name="emp1")
7. public class Employee implements Serializable {
8.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9.  
10. @Id  
11. @Column(name="ENO")  
12. private int eno;  
13. @Column(name="ENAME")  
14. private String ename;  
15. @Column(name="ESAL")  
16. private float esal;  
17. @Column(name="EADDR")  
18. private String eaddr;  
19. private static final long serialVersionUID = 1L;  
20.  
21. public Employee() {  
22.     super();  
23. }  
24. public int getEno() {  
25.     return this.eno;  
26. }  
27.  
28. public void setEno(int eno) {  
29.     this.eno = eno;  
30. }  
31. public String getEname() {  
32.     return this.ename;  
33. }  
34.  
35. public void setEname(String ename) {  
36.     this.ename = ename;  
37. }  
38. public float getEsal() {  
39.     return this.esal;  
40. }  
41.  
42. public void setEsal(float esal) {  
43.     this.esal = esal;  
44. }  
45. public String getEaddr() {  
46.     return this.eaddr;  
47. }  
48.  
49. public void setEaddr(String eaddr) {  
50.     this.eaddr = eaddr;  
51. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

52.
53.}

3. Create JPA configuration File

Open persistence.xml file which is existed under "src\META-INF" folder and provide the following details.

persistence.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
3.   <persistence-unit name="emp">
4.     <class>com.durgasoft.entity.Employee</class>
5.     <properties>
6.       <property name="javax.persistence.jdbc.driver" value="oracle.jdbc.OracleDriver"/>
7.       <property name="javax.persistence.jdbc.url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
8.       <property name="javax.persistence.jdbc.user" value="system"/>
9.       <property name="javax.persistence.jdbc.password" value="durga"/>
10.    </properties>
11.   </persistence-unit>
12. </persistence>
```

4. Create Test Application

Create a package "com.durgasoft.test" and prepare Test class:

Test.java

```
1. package com.durgasoft.test;
2.
3. import javax.persistence.EntityManager;
4. import javax.persistence.EntityManagerFactory;
5. import javax.persistence.EntityTransaction;
6. import javax.persistence.Persistence;
7.
8. import com.durgasoft.entity.Employee;
9.
10. public class Test {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
12. public static void main(String[] args) throws Exception {  
13. EntityManagerFactory factory = Persistence.createEntityManagerFactory("emp");  
14. EntityManager entityManager = factory.createEntityManager();  
15. /*  
16. EntityTransaction entityTransaction = entityManager.getTransaction();  
17. entityTransaction.begin();  
18. Employee emp = new Employee();  
19. emp.setEno(111);  
20. emp.setEname("AAA");  
21. emp.setEsal(5000);  
22. emp.setEaddr("Hyd");  
23. entityManager.persist(emp);  
24. entityTransaction.commit();  
25. System.out.println("Employee Inserted Successfully");  
26. */  
27. /*  
28. Employee emp = entityManager.find(Employee.class, 111);  
29. System.out.println("Employee Details");  
30. System.out.println("-----");  
31. System.out.println("Employee Number : "+emp.getEno());  
32. System.out.println("Employee Name : "+emp.getEname());  
33. System.out.println("Employee Salary : "+emp.getEsal());  
34. System.out.println("Employee Address : "+emp.getEaddr());  
35. */  
36. /*  
37. EntityTransaction entityTransaction = entityManager.getTransaction();  
38. entityTransaction.begin();  
39. Employee emp = entityManager.find(Employee.class, 111);  
40. emp.setEname("BBB");  
41. emp.setEsal(7000);  
42. emp.setEaddr("Sec");  
43. entityTransaction.commit();  
44. System.out.println("Employee updated Successfully");  
45. */  
46. EntityTransaction entityTransaction = entityManager.getTransaction();  
47. entityTransaction.begin();  
48. Employee emp = entityManager.find(Employee.class, 111);  
49. entityManager.remove(emp);  
50. entityTransaction.commit();  
51. System.out.println("Employee Deleted Successfully");  
52. entityManager.close();  
53. }  
54. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Integration of JPA with Spring application in ORM Module

Integration of JPA with Spring application in ORM Module:

1. Create Java Project with both Spring Library and Hibernate Library.
2. Create Dao interface
3. Create Dao implementation classs.
4. Create POJO / Entity class.
5. Create Hibernate Mapping File.
6. Create Spring Configuration File.
7. Create Test Application.

1. Create Java Project with both Spring Library and Hibernate Library.

Spring Library:

- ⊕ commons-logging-1.2.jar
- ⊕ spring-aop-4.3.9.RELEASE.jar
- ⊕ spring-beans-4.3.9.RELEASE.jar
- ⊕ spring-context-4.3.9.RELEASE.jar
- ⊕ spring-context-support-4.3.9.RELEASE.jar
- ⊕ spring-core-4.3.9.RELEASE.jar
- ⊕ spring-expression-4.3.9.RELEASE.jar
- ⊕ spring-jdbc-4.3.9.RELEASE.jar
- ⊕ spring-orm-4.3.9.RELEASE.jar
- ⊕ spring-tx-4.3.9.RELEASE.jar

Hibernate Library:

- ⊕ hibernate3.jar
- ⊕ antlr-2.7.6.jar
- ⊕ commons-collections-3.1.jar
- ⊕ dom4j-1.6.1.jar
- ⊕ javassist-3.12.0.GA.jar
- ⊕ jta-1.1.jar
- ⊕ slf4j-api-1.6.1.jar
- ⊕ hibernate-jpa-2.0-api-1.0.1.Final.jar
- ⊕ ojdbc6.jar

2. Create Dao interface

EmployeeDao.java

```
1. package com.durgasoftware.dao;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftwareonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2.
3. import com.durgasoft.entity.Employee;
4.
5. public interface EmployeeDao {
6.     public String insertEmployee(Employee emp);
7.     public Employee findEmployee(int eno);
8.     public String updateEmployee(Employee emp);
9.     public String removeEmployee(int eno);
10.}
```

3.Create Dao implementation class



```

1. package com.durgasoft.dao;
2.
3. import javax.persistence.EntityManager;
4. import javax.persistence.PersistenceContext;
5.
6. import org.springframework.stereotype.Repository;
7. import org.springframework.transaction.annotation.Transactional;
8.
9. import com.durgasoft.entity.Employee;
10.
11. @Repository
12. public class EmployeeDaolmpl implements EmployeeDao {
13.
14.     String status = "";
15.
16.     @PersistenceContext
17.     private EntityManager entityManager;
18.
19.     @Transactional
20.     @Override
21.     public String insertEmployee(Employee emp) {
22.         entityManager.persist(emp);
23.         status = "SUCCESS";
24.         return status;
25.     }
26.
27.     @Override
28.     public Employee findEmployee(int eno) {
29.         Employee emp = (Employee)entityManager.find(Employee.class, eno);
30.         return emp;
31.     }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

32.
33. @Transactional
34. @Override
35. public String updateEmployee(Employee emp) {
36.     Employee employee = (Employee)entityManager.find(Employee.class, emp.getEno())
37.     ;
38.     employee.setEname(emp.getEname());
39.     employee.setEsal(emp.getEsal());
40.     employee.setEaddr(emp.getEaddr());
41.     status = "SUCCESS";
42.     return status;
43.
44. @Transactional
45. @Override
46. public String removeEmployee(int eno) {
47.     Employee employee = (Employee)entityManager.find(Employee.class, eno);
48.     entityManager.remove(employee);
49.     status = "SUCCESS";
50.     return status;
51. }
52.
53.}
```

- ✓ Where "@PersistenceContext" will inject EntityManager object into Dao implementation class.
- ✓ Where "@Transactional" annotation will inject Transaction service in DAO methods, where it is not required to Create EntityTransaction object and not required to perform commit() or rollback() operations.

4. Create POJO / Entity class:

Employee.java

```

1. package com.durgasoft.entity;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12. public void setEno(int eno) {
13.     this.eno = eno;
14. }
15. public String getEname() {
16.     return ename;
17. }
18. public void setEname(String ename) {
19.     this.ename = ename;
20. }
21. public float getEsal() {
22.     return esal;
23. }
24. public void setEsal(float esal) {
25.     this.esal = esal;
26. }
27. public String getEaddr() {
28.     return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.     this.eaddr = eaddr;
32. }
33.

```

5. Create Hibernate Mapping File.

Employee.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.entity.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
11.  </class>
12. </hibernate-mapping>

```

6. Create Spring Configuration File

applicationContext.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:aop="http://www.springframework.org/schema/aop"
5.   xmlns:tx="http://www.springframework.org/schema/tx"
6.   xmlns:context="http://www.springframework.org/schema/context"
7.
8.   xsi:schemaLocation="
9.     http://www.springframework.org/schema/beans
10.    http://www.springframework.org/schema/beans/spring-beans.xsd
11.    http://www.springframework.org/schema/tx
12.    http://www.springframework.org/schema/tx/spring-tx.xsd
13.    http://www.springframework.org/schema/aop
14.    http://www.springframework.org/schema/aop/spring-aop.xsd
15.    http://www.springframework.org/schema/context
16.    http://www.springframework.org/schema/context/spring-context.xsd">
17. <context:annotation-config/>
18. <tx:annotation-driven/>
19. <bean id="dataSource"  class="org.springframework.jdbc.datasource.DriverManagerData
Source">
20.   <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
21.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
22.   <property name="username" value="system"/>
23.   <property name="password" value="durga"/>
24. </bean>
25. <bean id="entityManagerFactoryBean" class="org.springframework.orm.jpa.LocalContain
rEntityManagerFactoryBean">
26.   <property name="dataSource" ref="dataSource"/>
27.   <property name="persistenceUnitName" value="emp"/>
28.   <property name="jpaVendorAdapter">
29.     <bean class="org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter"/>
30.   </property>
31.   <property name="mappingResources">
32.     <list>
33.       <value>Employee.xml</value>
34.     </list>
35.   </property>
36.   <property name="jpaProperties">
37.     <props>
38.       <prop key="hibernate.dialect">org.hibernate.dialect.OracleDialect</prop>
39.       <prop key="hibernate.show_sql">true</prop>
40.     </props>
41.   </property>
42. </bean>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

43. <bean id="transactionManager" class="org.springframework.orm.jpa.JpaTransactionManager">
44.   <property name="entityManagerFactory" ref="entityManagerFactoryBean"/>
45. </bean>
46. <bean id="empDao" class="com.durgasoft.dao.EmployeeDaoImpl"/>
47. </beans>

```

- ✓ Where "org.springframework.jdbc.datasource.DriverManagerDataSource" will provide dataSource object, it required the following dependencies.
 1. driverClassName
 2. url
 3. username
 4. password
- ✓ Where "org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean" will provide EntityManager object in Spring Application and it require the following dependencies.
 1. dataSource
 2. persistenceUnitName
 3. jpaVendorAdapter
 4. mappingResources
 5. jpaProperties
- ✓ Where "dataSource" will take a DataSource reference which we configure in Spring configuration File.
- ✓ Where "persistenceUnitName" will take persistence name like "emp".
- ✓ Where "jpaVendorAdpter" property will take the class like "org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter" and it will provide all Hibernate implementation of JPA to Spring application.
- ✓ Where "mappingResources" property will take all mapping Files which we are using in spring applications in the form of <list> type.
- ✓ Where "jpaProperties" will take JPA implementation properties like dialect, show_sql,... in the form of "<props>" type.
- ✓ Where "org.springframework.orm.jpa.JpaTransactionManager" will provide Transaction implementation provided by JPA vendor.
- ✓ Where "<context:annotation-config/>" tag will activate the annotations like "@Repository", "@Service", "@Component"
- ✓ Where "<tx:annotation-driven/>" tag will activate @Transactional annotation which will inject Transaction Service in Spring application.

7. Create Test Application:

```

1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5. import org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter;
6.
7. import com.durgasoft.dao.EmployeeDao;
8. import com.durgasoft.entity.Employee;
9.
10. public class Test {
11.
12.     public static void main(String[] args) {
13.         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
14.         EmployeeDao empDao = (EmployeeDao) context.getBean("empDao");
15.         /*
16.         Employee emp = new Employee();
17.         emp.setEno(111);
18.         emp.setEname("AAA");
19.         emp.setEsal(5000);
20.         emp.setEaddr("Hyd");
21.         String status = empDao.insertEmployee(emp);
22.         System.out.println(status);
23.         */
24.         /*
25.         Employee emp = empDao.findEmployee(111);
26.         System.out.println("Employee Details");
27.         System.out.println("-----");
28.         System.out.println("Employee Number :" + emp.getEno());
29.         System.out.println("Employee Name :" + emp.getEname());
30.         System.out.println("Employee Salary :" + emp.getEsal());
31.         System.out.println("Employee Address :" + emp.getEaddr());
32.         */
33.         /*
34.         Employee emp = new Employee();
35.         emp.setEno(111);
36.         emp.setEname("BBB");
37.         emp.setEsal(7000);
38.         emp.setEaddr("Pune");
39.         String status = empDao.updateEmployee(emp);
40.         System.out.println(status);
41.         */
42.         String status = empDao.removeEmployee(111);
43.         System.out.println(status);
44.     }
45. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**Employee.java**

```
1. package com.durgasoft.pojo;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.     public void setEno(int eno) {
13.         this.eno = eno;
14.     }
15.     public String getEname() {
16.         return ename;
17.     }
18.     public void setEname(String ename) {
19.         this.ename = ename;
20.     }
21.     public float getEsal() {
22.         return esal;
23.     }
24.     public void setEsal(float esal) {
25.         this.esal = esal;
26.     }
27.     public String getEaddr() {
28.         return eaddr;
29.     }
30.     public void setEaddr(String eaddr) {
31.         this.eaddr = eaddr;
32.     }
33.
34.     public String toString() {
35.         return "["+eno+","+ename+","+esal+","+eaddr+"]";
36.     }
37.
38. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EmployeeDao.java

```
1. package com.durgasoft.dao;
2.
3. import com.durgasoft.entity.Employee;
4.
5. public interface EmployeeDao {
6.     public String insertEmployee(Employee emp);
7.     public Employee findEmployee(int eno);
8.     public String updateEmployee(Employee emp);
9.     public String removeEmployee(int eno);
10.}
```

EmployeeDaoImpl.java

```
1. package com.durgasoft.dao;
2.
3. import javax.persistence.EntityManager;
4. import javax.persistence.PersistenceContext;
5.
6. import org.springframework.stereotype.Repository;
7. import org.springframework.transaction.annotation.Transactional;
8.
9. import com.durgasoft.entity.Employee;
10.
11. @Repository
12. public class EmployeeDaoImpl implements EmployeeDao {
13.
14.     String status = "";
15.
16.     @PersistenceContext
17.     private EntityManager entityManager;
18.
19.     @Transactional
20.     @Override
21.     public String insertEmployee(Employee emp) {
22.         entityManager.persist(emp);
23.         status = "SUCCESS";
24.         return status;
25.     }
26.
27.     @Override
28.     public Employee findEmployee(int eno) {
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

29.     Employee emp = (Employee)entityManager.find(Employee.class, eno);
30.     return emp;
31. }
32.
33. @Transactional
34. @Override
35. public String updateEmployee(Employee emp) {
36.     Employee employee = (Employee)entityManager.find(Employee.class, emp.getEno())
37. ;
38.     employee.setEname(emp.getEname());
39.     employee.setEsal(emp.getEsal());
40.     employee.setEaddr(emp.getEaddr());
41.     status = "SUCCESS";
42.     return status;
43.
44. @Transactional
45. @Override
46. public String removeEmployee(int eno) {
47.     Employee employee = (Employee)entityManager.find(Employee.class, eno);
48.     entityManager.remove(employee);
49.     status = "SUCCESS";
50.     return status;
51. }
52.
53.

```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Employee" table="emp2">
7.     <id name="eno" column="ENO"/>
8.     <property name="ename" column="ENAME"/>
9.     <property name="esal" column="ESAL"/>
10.    <property name="eaddr" column="EADDR"/>
11.  </class>
12. </hibernate-mapping>

```

applicationContext.xml

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:aop="http://www.springframework.org/schema/aop"
5.   xmlns:tx="http://www.springframework.org/schema/tx"
6.   xmlns:context="http://www.springframework.org/schema/context"
7.
8.   xsi:schemaLocation="
9.     http://www.springframework.org/schema/beans
10.    http://www.springframework.org/schema/beans/spring-beans.xsd
11.    http://www.springframework.org/schema/tx
12.    http://www.springframework.org/schema/tx/spring-tx.xsd
13.    http://www.springframework.org/schema/aop
14.    http://www.springframework.org/schema/aop/spring-aop.xsd
15.    http://www.springframework.org/schema/context
16.    http://www.springframework.org/schema/context/spring-context.xsd">
17. <context:annotation-config/>
18. <tx:annotation-driven/>
19. <bean id="dataSource"  class="org.springframework.jdbc.datasource.DriverManagerData
Source">
20.   <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
21.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
22.   <property name="username" value="system"/>
23.   <property name="password" value="durga"/>
24. </bean>
25. <bean id="entityManagerFactoryBean" class="org.springframework.orm.jpa.LocalContain
rEntityManagerFactoryBean">
26.   <property name="dataSource" ref="dataSource"/>
27.   <property name="persistenceUnitName" value="emp"/>
28.   <property name="jpaVendorAdapter">
29.     <bean class="org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter"/>
30.   </property>
31.   <property name="mappingResources">
32.     <list>
33.       <value>Employee.xml</value>
34.     </list>
35.   </property>
36.   <property name="jpaProperties">
37.     <props>
38.       <prop key="hibernate.dialect">org.hibernate.dialect.OracleDialect</prop>
39.       <prop key="hibernate.show_sql">true</prop>
40.     </props>
41.   </property>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
42.</bean>
43.<bean id="transactionManager" class="org.springframework.orm.jpa.JpaTransactionManager">
44.  <property name="entityManagerFactory" ref="entityManagerFactoryBean"/>
45.</bean>
46.<bean id="empDao" class="com.durgasoft.dao.EmployeeDaoImpl"/>
47.</beans>
```

**Test.java**

```
1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5. import org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter;
6.
7. import com.durgasoft.dao.EmployeeDao;
8. import com.durgasoft.entity.Employee;
9.
10. public class Test {
11.
12.     public static void main(String[] args) {
13.         ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
14.         EmployeeDao empDao = (EmployeeDao) context.getBean("empDao");
15.         /*
16.         Employee emp = new Employee();
17.         emp.setEno(111);
18.         emp.setEname("AAA");
19.         emp.setEsal(5000);
20.         emp.setEaddr("Hyd");
21.         String status = empDao.insertEmployee(emp);
22.         System.out.println(status);
23.         */
24.         /*
25.         Employee emp = empDao.findEmployee(111);
26.         System.out.println("Employee Details");
27.         System.out.println("-----");
28.         System.out.println("Employee Number : "+emp.getEno());
29.         System.out.println("Employee Name : "+emp.getEname());
30.         System.out.println("Employee Salary : "+emp.getEsal());
31.         System.out.println("Employee Address : "+emp.getEaddr());
32.         */
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
33.  /*
34.  Employee emp = new Employee();
35.  emp.setEno(111);
36.  emp.setEname("BBB");
37.  emp.setEsal(7000);
38.  emp.setEaddr("Pune");
39.  String status = empDao.updateEmployee(emp);
40.  System.out.println(status);
41.  */
42.  String status = empDao.removeEmployee(111);
43.  System.out.println(status);
44. }
45.
46.}
```

DURGAGAUSY

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGA SOFT { SPRING TRANSACTIONS }

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING TRANSACTIONS MODULE INDEX

- | | |
|---|----------|
| 1. Intorduction..... | PAGE 406 |
| 2. Transaction Support in Spring..... | PAGE 411 |
| 3. Transaction Attributes..... | PAGE 412 |
| 4. Transactions Approaches in Spring..... | PAGE 414 |

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring Transactions

Introduction:

Transaction Management:

Transaction is an unit of work performed by Front End applications on Back End System.

EX: Deposit some amount in an Account.

Withdraw some amount from an Account

Transfer some amount from one account to another account.

IN database applications, Every Transaction must follow ACID properties

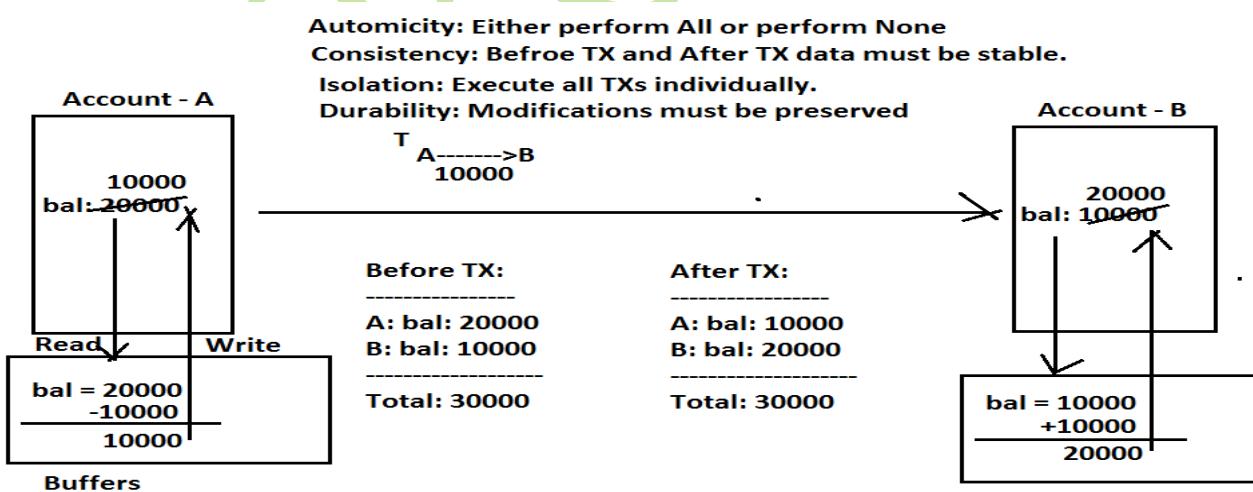
1. Atomicity: This property will make the Transaction either in SUCCESS state or in FAILURE state.

In Database related applications, if we perform all operations then the Transaction is available in SUCCESS State, if we perform none of the operations then the Transaction is available in FAILURE state.

2. Consistency: In database applications, Before the Transaction and After the Transaction Database state must be in stable.

3. Isolation: If we run more than one Transaction on a single Data item then that Transactions are called as "Concurrent Transactions". In Transactions Concurrency , one transaction execution must not give effect to another Transaction, this rule is called as "Isolation" property.

4. Durability: After committing the Transaction, if any failures are coming like Power failure, OS failure,... after getting the System if we open the transaction then the modifications which we performed during the transaction must be preserved.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In JDBC, to perform Automicity property we have to change Connections auto-commit nature and we have to perform either commit() or rollback() at the end of Transaction.

EX:

```

1. Connection con = DriverManager.getConnection(---);
2. con.setAutoCommit(false);
3. try{
4. ---instructions-----
5. con.commit();
6. }catch(Exception e){
7. e.printStackTrace();
8. con.rollback();
9. }
```

In Hibernate applications, if we want to manage Transactions Automicity property then we have to use the following steps.

1. Declare Transaction Before try.
2. Create Transaction object inside try block.
3. Perform commit() operation at end of Transaction.
4. Perform rollback() operation at catch block.

Transaction tx = null;

```

1. try{
2. ----
3. tx = session.beginTransaction();
4. ----
5. ----
6. tx.commit();
7. }catch(Exception e){
8. tx.rollback();
9. }
```

If we execute more than one transaction on a single data item then that transactions are called as Concurrent Transactions.

In Transactions concurrency we are able to get the following data consistency problems while executing more than one transaction at a time.

1. Lost Update Problem
2. Dirty Read Problem
3. Non Repeatable Read Problem
4. Phantasm Read Problem

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

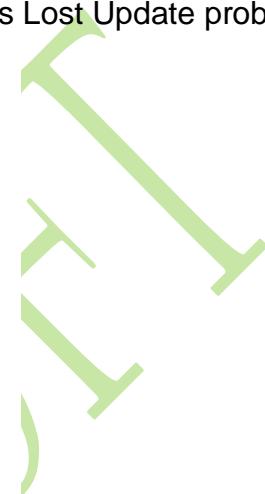
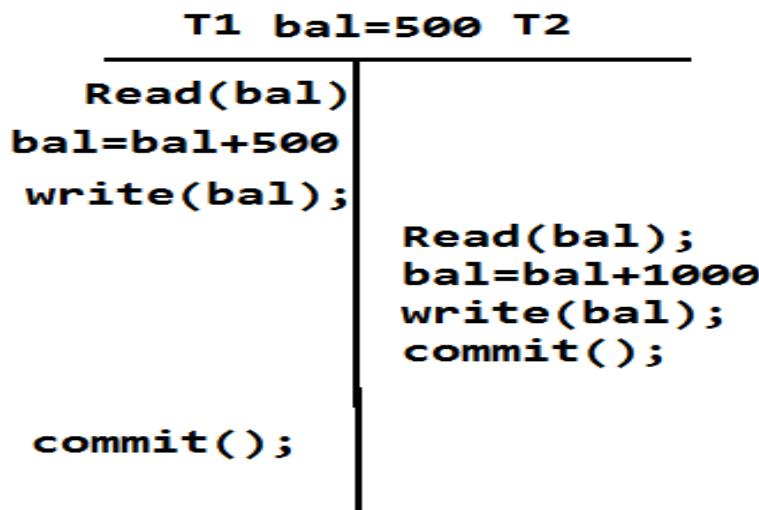
FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

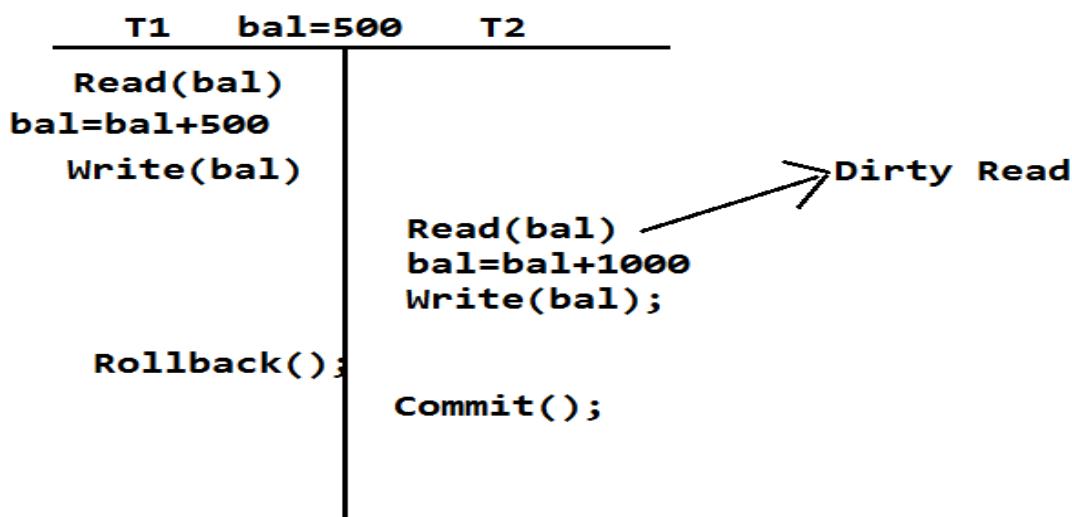
1. Lost Update Problem

In Transactions concurrency, if one transaction perform updatations over the data with out commit operation , mean while, other transactions perform updatations with commit operation then the first transaction updatations are lost, this data consistency problem is called as Lost Update problem.



2. Dirty Read Problem:

In Transactions concurrency, if one transaction perform updatations over data with out performing commit / rollback, mean while if other Transaction perform Read operation over the uncommitted data with out performing commit/rollback operations, in this context, if first transaction perform Rollback operation then the read operation performed by second transaction is Dirty Read, this problem is called as Dirty Read problem.



Dirty Read

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

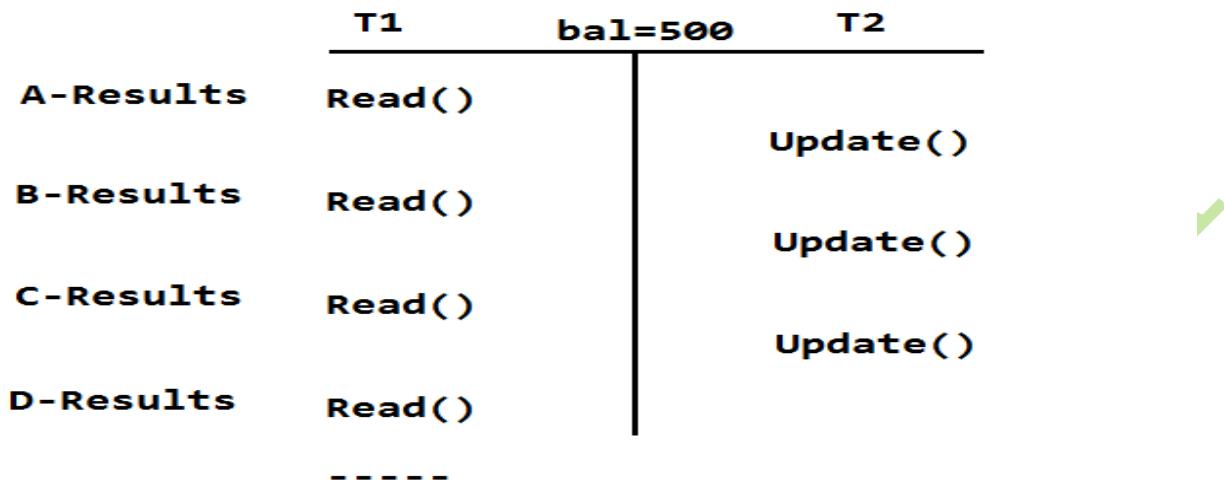
FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

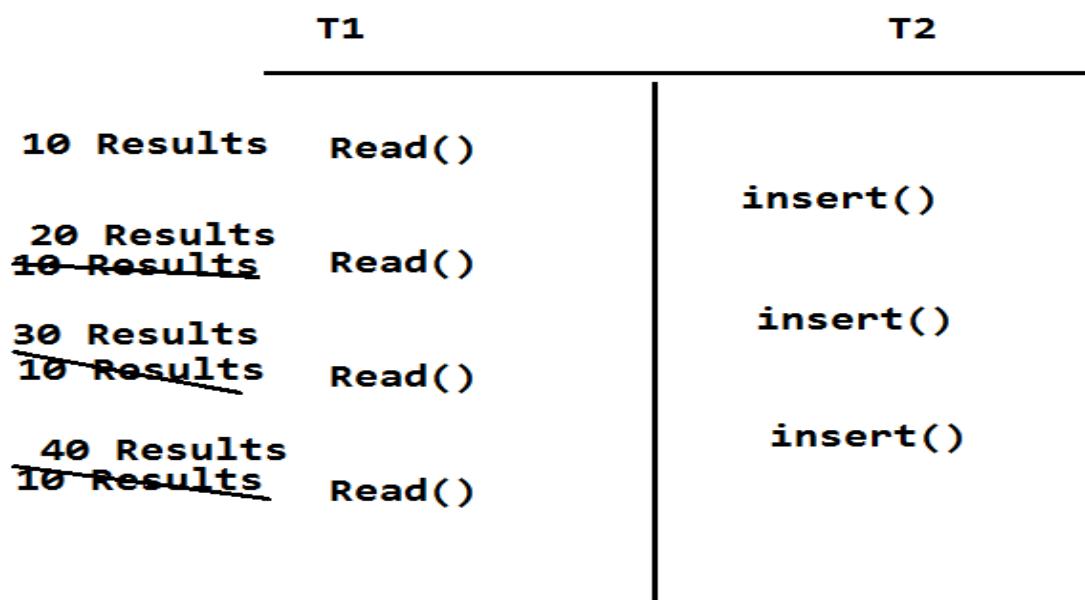
3. Non Repeatable Read Problem

In Transactions concurrency, One transaction perform continuous read operations to get same results, mean while, between two read operations another transaction performs update operation over the same data, in this context, in the next read operation performed by first transaction may not get same repeatable results, this problem is called as Non Repeatable Read Problem.



4. Phantom Read Problem

In Transactions concurrency, one transaction perform read operation continuously to get same no of results at each and every read operation , mean while, other transactions may perform insert operations between two read operations performed by first transactions, in this context, in the next read operation performed by first transaction may not generate the same no of results, this problem is called as "Panphant Read" Problem, here the extra records inserted by second transaction are called as "Panphant Records".



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. public static final int TRANSACTION_NONE = 0;
2. public static final int TRANSACTION_READ_UNCOMMITTED = 1;
3. public static final int TRANSACTION_READ_COMMITTED = 2;
4. public static final int TRANSACTION_REPEATABLE_READ = 4;
5. public static final int TRANSACTION_SERIALIZABLE = 8;

```
public void setTransactionIsolation(int isolation_Level)
```

EX: con.setTransactionIsolation(Connection.TRANSACTION_READ_COMMITTED);

<property name="hibernate.connection.isolation>val</property>

Where value may be either of the following Constants

NONE = 0;
READ_UNCOMMITTED = 1;
READ_COMMITTED = 2;
REPEATABLE_READ = 4;
SERIALIZABLE = 8;

EX:

1. <hibernate-configuration>
2. <session-factory>
3. ----
4. <property name="hibernate.connection.isolation>
5. SERIALIZABLE
6. </property>
7. ----
8. </session-factory>
9. </hibernate-configuration>

There are two types of Transactions

1. Local Transaction
2. Global Transaction

1. Local Transaction

Local transactions are specific to a single transactional resource like a JDBC connection

Local transaction management can be useful in a centralized computing environment where application components and resources are located at a single site, and transaction management only involves a local data manager running on a single machine. Local transactions are easier to be implemented

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Global Transaction

Global transactions can span multiple transactional resources like transaction in a distributed system

Global transaction management is required in a distributed computing environment where all the resources are distributed across multiple systems

A distributed or a global transaction is executed across multiple systems, and its execution requires coordination between the global transaction management system and all the local data managers of all the involved systems.

Transaction Support in Spring

Spring provides extensive support for transaction management and help developers to focus more on business logic rather than worrying about the integrity of data incase of any system failures.

Spring Transaction Management is providing the following advantages in enterprise applications:

1. Spring Supports Declarative Transaction Management. In this model, Spring uses AOP over the transactional methods to provide data integrity. This is the preferred approach and works in most of the cases.
2. Spring Supports most of the transaction APIs such as JDBC, Hibernate, JPA, JDO, JTA etc. All we need to do is use proper transaction manager implementation class.

EX:

- org.springframework.jdbc.datasource.DriverManagerDataSource for JDBC transaction management
- org.springframework.orm.hibernate3.HibernateTransactionManager for Hibernate as ORM tool.
- Support for programmatic transaction management by using TransactionTemplate or PlatformTransactionManager implementation.

To represent ISOLATION levels Spring Framework has provided the following Constants from "org.springframework.transaction .TransactionDefinition".

1. **ISOLATION_DEFAULT:** It is default isolation level, it will use the underlying database provided default Isolation level.
2. **ISOLATION_READ_UNCOMMITTED:** It will not resolve any ISOLATION problem, It represents dirty read problem, non-repeatable read problem, phantom read problem .
3. **ISOLATION_READ_COMMITTED:** It will resolve dirty read problem, but, it will represent non-repeatable read problem and phantom read problem.
4. **ISOLATION_REPEATABLE_READ:** It will resolve dirty read problem and non-repeatable read problem , but, It will represent phantom read problem.
5. **ISOLATION_SERIALIZABLE:** It will resolve all ISOLATION problems like dirty reads, non-repeatable read, and phantom read Problems.

To set the above Isolation level to TransactionTemplate then we have to use the following method
.public void setIsolationLevel(int value)

EX: txTemplate.setIsolationLevel(TransactionDefinition.ISOLATION_DEFAULT);

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Transaction Attributes

In Enterprise applications, if we a method begins a transaction and access another method, in this context, another method execution is going on in the same Transaction or in any new Transaction is completely depending on the Transaction Propagation behaviour what we are using.

Spring Framework has provided very good support for Propagation Behaviour in the form of the following constants from "org.springframework.transaction .TransactionDefinition".

1. PROPAGATION_REQUIRED
2. PROPAGATION_REQUIRES_NEW
3. PROPAGATION_SUPPORTS
4. PROPAGATION_NOT_SUPPORTED
5. PROPAGATION_MANDATORY
6. PROPAGATION_NEVER
7. PROPAGATION_NESTED

1. PROPAGATION_REQUIRED:

If a method1 is running in a transaction and invokes method2 then method2 will be executed in the same transaction. If method1 is not associated with any Transaction then Container will create new Transaction to execute Method2.

2. PROPAGATION_REQUIRES_NEW:

If a method1 is running in a transaction and invokes method2 then container will suspend the current Transaction temporarily and creates new Transaction for method2. After executing method2 transaction then method1 transaction will continue. If Method1 is not associated with any transaction then container will start a new transaction before starts new method.

3. PROPAGATION_MANDATORY:

If a method1 is running in a transaction and invokes method2 then method2 will be executed in the same transaction. If method1 is not associated with any Transaction then Container will raise an exception like "TransactionThrowsException".

4. PROPAGATION_SUPPORTS:

If a method1 is running in a transaction and invokes method2 then method2 will be executed in the same transaction. If method1 is not associated with any Transaction then Container does not start new Transaction before running method2.

5. PROPAGATION_NOT_SUPPORTED:

If a method1 is running in a transaction and invokes method2 then Container Suspends the Method1 transaction before invoking Method2. When Method2 has completed, container resumes Method1 transaction. If Method1 is not associated with Transaction then Container does not start new Transaction before executing Method2.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

6. PROPAGATION_NEVER:

If a method1 is running in a transaction and invokes method2 then Container throws an Exception like RemoteException. If method1 is not associated with any transaction then Container will not start new Transaction for Method2.

7. PROPAGATION_NESTED:

Indicates that the method should be run with in a nested transaction if an existed transaction is in progress.

To set the above Propagation Behaviour to TransactionTemplate then we have to use the following method.

```
public void setPropagationBehaviour(int value)
```

EX:

```
txTemplate.setPropagationBehaviour(TransactionDefinition.PROPAGATION_EQUIRES_NEW);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftware.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Transactions Approaches in Spring

Spring Framework supports Transactions in two ways.

1. Programmatic Approach
2. Declarative Approach

Programmatic Approach:

In Programmatic Approach of transactions we have to declare the transaction and we have to perform transactions commit or rollback operations explicitly by using JAVA code.

In Programmatic approach if we want to provide transactions then we have to use the following predefined Library.

1. Transaction Manager:

The main intention of TransactionManager is able to define a Transaction Strategy in spring application.

Spring Framework has provided Transaction manager in the form of a predefined interfaces

1. org.springframework.jdbc.datasource.DataSourceTransactionManager
 2. org.springframework.transaction.PlatformTransactionManager
 3. org.springframework.orm.hibernate4.HibernateTransactionManager
 4. org.springframework.transaction.jta.JtaTransactionManager
- -----

In Spring Transaction based applications , We must configure either of the above TransactionManager in configuration file and we must inject TransactionManager in DAO implementation class.

DataSourceTransactionManager includes the following methods to manage transaction.

1. public TransactionStatus getTransaction(TransactionDefinition tx_Def)
2. public void commit(TransactionStatus tx_Status)
3. public void rollback(TransactionStatus tx_Status)

2. TransactionDefinition:

org.springframework.transaction.TransactionDefinition is able to specify ISOLATION levels, Propagation behaviours, Transactions Time out and Transactions Read Only status,.....

TransactionDefinition includes the following Constants to manage Transactions ISOLATION Levels in Spring applications.

1. ISOLATION_READ_UNCOMMITTED

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. ISOLATION_READ_COMMITTED
3. ISOLATION_REPEATABLE_READ
4. ISOLATION_SERIALIZABLE

TransactionDefinition includes the following Constants to manage Transactions Propagation Behaviours in Spring applications.

1. PROPAGATION_REQUIRED
2. PROPAGATION.Requires_NEW
3. PROPAGATION_SUPPORTS
4. PROPAGATION_NOT_SUPPORTED
5. PROPAGATION_MANDATORY
6. PROPAGATION_NESTED
7. PROPAGATION_NEVER

3. TransactionStatus:

org.springframework.transaction.TransactionStatus interface provides a simple way for transactional code to control transaction execution and query transaction status.

TransactionStatus includes the following methods to manage Transactions in Spring applications.

1. public boolean isNewTransaction()
2. boolean hasSavepoint()
3. public void setRollbackOnly()
4. public boolean isRollbackOnly()
5. public void flush()
6. public boolean isCompleted()

Example on Spring Transactions in Programmatic Approach:

TransactionDao.java

```
1. package com.durgasoft.dao;
2. public interface TransactionDao {
3.     public String transferFunds(String fromAccount, String toAccount, int transfer_Amt);
4. }
```

TransactionDaolmpl.java

```
1. package com.durgasoft.dao;
2. import org.springframework.jdbc.core.JdbcTemplate;
3. import org.springframework.jdbc.datasource.DataSourceTransactionManager;
4. import org.springframework.transaction.TransactionDefinition;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. import org.springframework.transaction.TransactionStatus;
6. import org.springframework.transaction.support.DefaultTransactionDefinition;
7.
8. public class TransactionDaoImpl implements TransactionDao {
9.
10.    private JdbcTemplate jdbcTemplate;
11.    private DataSourceTransactionManager transactionManager;
12.
13.    public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
14.        this.jdbcTemplate = jdbcTemplate;
15.    }
16.    public void setTransactionManager(DataSourceTransactionManager transactionManager) {
17.        this.transactionManager = transactionManager;
18.    }
19.
20.    @Override
21.    public String transferFunds(String fromAccount, String toAccount, int transfer_Amt) {
22.        String status = "";
23.        TransactionDefinition tx_Def = new DefaultTransactionDefinition();
24.        TransactionStatus tx_Status = transactionManager.getTransaction(tx_Def);
25.        try {
26.            withdraw(fromAccount, transfer_Amt);
27.            deposit(toAccount, transfer_Amt);
28.            transactionManager.commit(tx_Status);
29.            status = "Transaction Success";
30.
31.        } catch (Exception e) {
32.            transactionManager.rollback(tx_Status);
33.            status = "Transaction Failure";
34.            e.printStackTrace();
35.        }
36.        return status;
37.    }
38.    public void withdraw(String acc, int wd_Amt) {
39.
40.        jdbcTemplate.execute("update account set BALANCE = BALANCE -
41.        "+wd_Amt+" where ACCNO = '"+acc+"'");
42.
43.    }
44.    public void deposit(String acc, int dep_Amt) throws Exception {
45.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

46.     float f = 100/0;
47.     jdbcTemplate.execute("update account set BALANCE = BALANCE + "+dep_Amt+
  where ACCNO = '"+acc+"'");
48.
49. }
50.

```

applicationContext.xml


```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:context="http://www.springframework.org/schema/context"
5.   xsi:schemaLocation="
6.     http://www.springframework.org/schema/beans
7.     http://www.springframework.org/schema/beans/spring-beans.xsd
8.     http://www.springframework.org/schema/context
9.     http://www.springframework.org/schema/context/spring-context.xsd">
10.  <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerData
  Source">
11.    <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
12.    <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
13.    <property name="username" value="system"/>
14.    <property name="password" value="durga"/>
15.  </bean>
16.  <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
17.    <property name="dataSource" ref="dataSource"/>
18.  </bean>
19.  <bean id="transactionDao" class="com.durgasoft.dao.TransactionDaoImpl">
20.    <property name="jdbcTemplate" ref="jdbcTemplate"/>
21.    <property name="transactionManager" ref="transactionManager"/>
22.  </bean>
23.  <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSource
  TransactionManager">
24.    <property name="dataSource" ref="dataSource"/>
25.  </bean>
26.
27.</beans>

```


Test.java

```

1. package com.durgasoft.test;
2.

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5.
6. import com.durgasoft.dao.TransactionDao;
7.
8. public class Test {
9.
10.    public static void main(String[] args) {
11.        ApplicationContext context = new ClassPathXmlApplicationContext("applicationConte
xt.xml");
12.        TransactionDao tx_Dao = (TransactionDao)context.getBean("transactionDao");
13.        String status = tx_Dao.transferFunds("abc123", "xyz123", 5000);
14.        System.out.println(status);
15.    }
16.}
```

Drawbacks with Transactions Programmatic Approach:

1. Programmatic Approach is suggestible when we have less no of operations in Transactions, it is not suggestible when we have more no of operations in Transactions.
2. Programmatic Approach is some what tightly coupled approach , because, it includes both Business logic and Transactions service code as a single unit.
3. Programmatic approach is not convenient to use in enterprise Applications.

Declarative Approach:

In Declarative approach, we will separate Transaction Service code and Business Logic in Spring applications , so that, we are able to get loosely coupled design in Spring applications.

Declarative approach is suggestible when we have more no of operations in Transactions.

Declarative approach is not convenient to use in enterprise Applications because of AOP.

There are two ways to provide Transaction management in declarative approach.

1. Using Transactions Namespace Tags with AOP implementation
2. Using Annotations.

1. Using Transactions Namespace Tags with AOP implementation

To manage Transactions in declarative approach, Spring Transaction module has provided the following AOP implemented tags.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. <tx:advice>

It represent Transaction Advice, it is the implementation of Transaction Service.

Syntax:

```
<tx:advice id="---" transaction-manager="----">
```

2. <tx:attributes>

It will take Transactional methods inorder to apply Isolation levels and Propagation behaviours,..... by using <tx:method> tag.

Syntax:

```
<tx:attributes>  
-----  
<tx:attributes>
```

3. <tx:method>

It will define Transactional method and its propagation Behaviours, Isolation levels, Timeout statuses,....

Syntax:

```
<tx:method name="---" propagation="---" isolation="-----" />
```

With the above tags, we must define Transaction Advice and it must be configured with a particular Pointcut expression by using <aop:advisor> tag.

EX:

```
1. <tx:advice id="txAdvice" transaction-manager="transactionManager">  
2.   <tx:attributes>  
3.     <tx:method name="transferFunds"/>  
4.   </tx:attributes>  
5. </tx:advice>  
6. <aop:config>  
7.   <aop:pointcut expression="execution(* com.durgasoft.dao.TransactionDao.transferFund  
s(..))" id="transfer"/>  
8.   <aop:advisor pointcut-ref="transfer" advice-ref="txAdvice"/>  
9. </aop:config>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

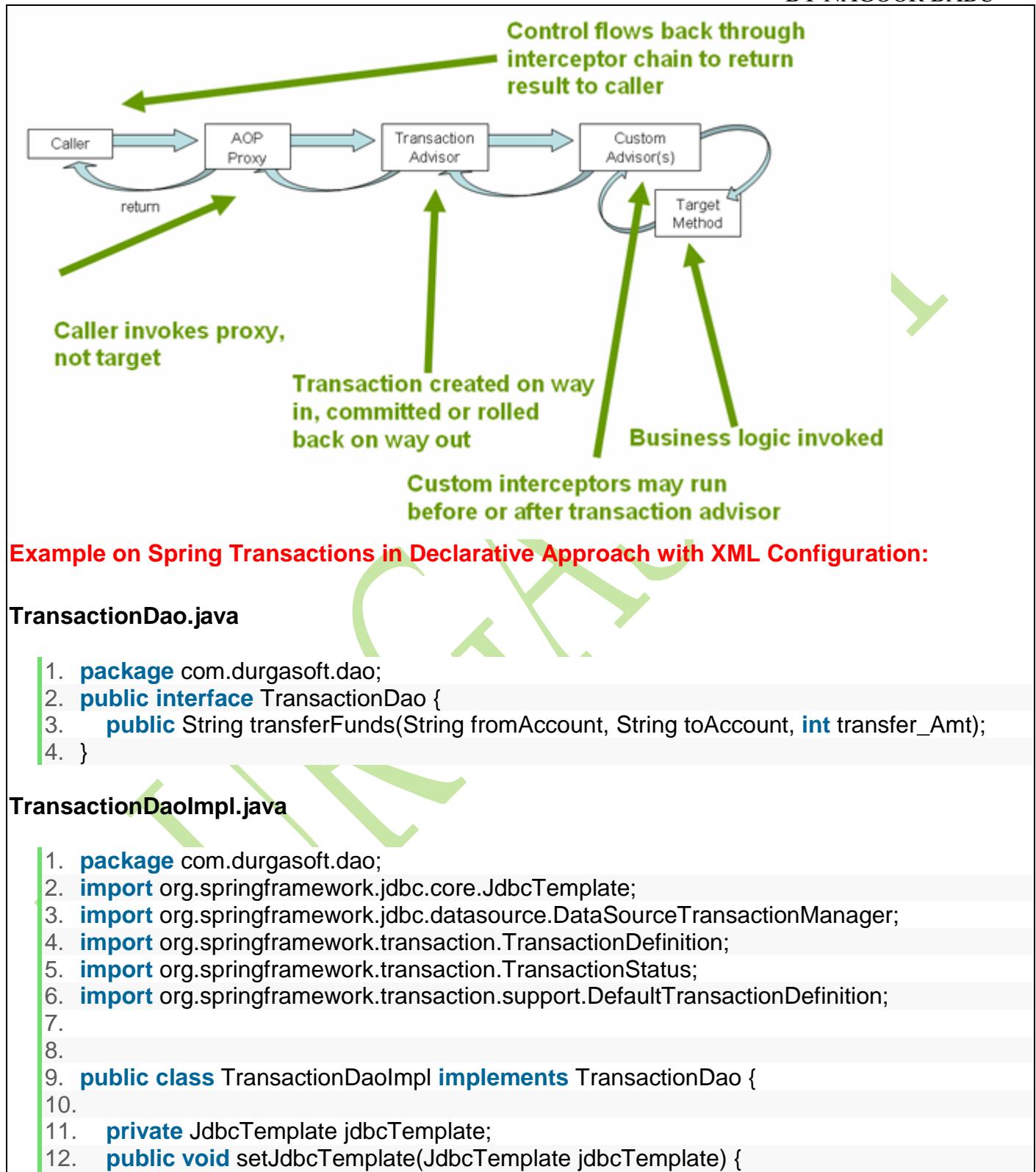
Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

13.     this.jdbcTemplate = jdbcTemplate;
14. }
15. @Override
16. public String transferFunds(String fromAccount, String toAccount, int transfer_Amt) {
17.     String status = "";
18.     int val1 = jdbcTemplate.update("update account set BALANCE = BALANCE -
19.         "+transfer_Amt+" where ACCNO = "+fromAccount+"");
20.     float f = 100/0;
21.     int val2 = jdbcTemplate.update("update account set BALANCE = BALANCE + "+tra-
22.         nsfer_Amt+" where ACCNO = "+toAccount+"");
23.     if(val1 ==1 && val2 ==1) {
24.         status = "Transaction Success";
25.     }else {
26.         status = "Transaction Failure";
27.     }
28. }
```

ApplicationContext.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:aop="http://www.springframework.org/schema/aop"
5.   xmlns:tx="http://www.springframework.org/schema/tx"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/tx
10.    http://www.springframework.org/schema/tx/spring-tx.xsd
11.    http://www.springframework.org/schema/aop
12.    http://www.springframework.org/schema/aop/spring-aop.xsd">
13.
14.
15. <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerData-
16. Source">
17.   <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
18.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
19.   <property name="username" value="system"/>
20.   <property name="password" value="durga"/>
21. <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

22.   <property name="dataSource" ref="dataSource"/>
23. </bean>
24. <bean id="transactionDao" class="com.durgasoft.dao.TransactionDaoImpl">
25.   <property name="jdbcTemplate" ref="jdbcTemplate"/>
26. </bean>
27. <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
28.   <property name="dataSource" ref="dataSource"/>
29. </bean>
30.
31. <tx:advice id="txAdvice" transaction-manager="transactionManager">
32.   <tx:attributes>
33.     <tx:method name="transferFunds"/>
34.   </tx:attributes>
35. </tx:advice>
36. <aop:config>
37.   <aop:pointcut expression="execution(* com.durgasoft.dao.TransactionDao.transferFunds(..))" id="transfer"/>
38.   <aop:advisor pointcut-ref="transfer" advice-ref="txAdvice"/>
39. </aop:config>
40.
41.</beans>

```

Test.java

```

1. package com.durgasoft.test;
2. import org.springframework.context.ApplicationContext;
3. import org.springframework.context.support.ClassPathXmlApplicationContext;
4.
5. import com.durgasoft.dao.TransactionDao;
6.
7. public class Test {
8.
9.   public static void main(String[] args) {
10.     ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
11.     TransactionDao tx_Dao = (TransactionDao)context.getBean("transactionDao");
12.     String status = tx_Dao.transferFunds("abc123", "xyz123", 100);
13.     System.out.println(status);
14.   }
15. }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Using Annotations:

This approach is very simple to use for transactions in Spring applications. In this approach, we will use `@Transactional` annotation just before the transactional methods in DAO implementation classes, but, to use this annotation we must activate `@Transactional` annotation in Spring configuration file by using the following tag.

```
<tx:annotation-driven transaction-manager="transactionManager"/>
```

Example on Spring Transactions in Declarative Approach with Annotations:

TransactionDao.java

```
1. package com.durgasoft.dao;
2. public interface TransactionDao {
3.     public String transferFunds(String fromAccount, String toAccount, int transfer_Amt);
4. }
```

TransactionDaolmpl.java

```
1. package com.durgasoft.dao;
2. import org.springframework.jdbc.core.JdbcTemplate;
3. import org.springframework.jdbc.datasource.DataSourceTransactionManager;
4. import org.springframework.transaction.TransactionDefinition;
5. import org.springframework.transaction.TransactionStatus;
6. import org.springframework.transaction.annotation.Transactional;
7. import org.springframework.transaction.support.DefaultTransactionDefinition;
8.
9.
10. public class TransactionDaolmpl implements TransactionDao {
11.
12.     private JdbcTemplate jdbcTemplate;
13.     public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {
14.         this.jdbcTemplate = jdbcTemplate;
15.     }
16.
17.     @Transactional
18.     @Override
19.     public String transferFunds(String fromAccount, String toAccount, int transfer_Amt) {
20.         String status = "";
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

21.     int val1 = jdbcTemplate.update("update account set BALANCE = BALANCE -
22.           "+transfer_Amt+" where ACCNO = "+fromAccount+"");
23.     float f = 100/0;
24.     int val2 = jdbcTemplate.update("update account set BALANCE = BALANCE + " + tra-
25.           nsfer_Amt+ " where ACCNO = "+toAccount+"");
26.     if(val1 ==1 && val2 ==1) {
27.         status = "Transaction Success";
28.     } else {
29.         status = "Transaction Failure";
30.     }
31. }
```

ApplicationContext.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:aop="http://www.springframework.org/schema/aop"
5.   xmlns:tx="http://www.springframework.org/schema/tx"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/tx
10.    http://www.springframework.org/schema/tx/spring-tx.xsd
11.    http://www.springframework.org/schema/aop
12.    http://www.springframework.org/schema/aop/spring-aop.xsd">
13.
14. <tx:annotation-driven transaction-manager="transactionManager"/>
15. <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerData-
Source">
16.   <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
17.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
18.   <property name="username" value="system"/>
19.   <property name="password" value="durga"/>
20. </bean>
21. <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
22.   <property name="dataSource" ref="dataSource"/>
23. </bean>
24. <bean id="transactionDao" class="com.durgasoft.dao.TransactionDaoImpl">
25.   <property name="jdbcTemplate" ref="jdbcTemplate"/>
26. </bean>
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
27. <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
28.   <property name="dataSource" ref="dataSource"/>
29. </bean>
30.
31.
32.</beans>
```

Test.java

```
1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5.
6. import com.durgasoft.dao.TransactionDao;
7.
8. public class Test {
9.
10.    public static void main(String[] args) {
11.        ApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");
12.        TransactionDao tx_Dao = (TransactionDao)context.getBean("transactionDao");
13.        String status = tx_Dao.transferFunds("abc123", "xyz123", 100);
14.        System.out.println(status);
15.    }
16.}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

{SPRING WEB MODULE}

DURGA SOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING WEB MODULE INDEX

1.	Intorduction.....	PAGE 428
2.	Steps to prepare Spring web MVC Application.....	PAGE 446
3.	Annotations in Spring WEB MVC.....	PAGE 461
4.	Spring Web MVC Application with MAVEN.....	PAGE 468
5.	Controller Classes.....	PAGE 473
6.	Data Validations.....	PAGE 542
7.	HandlerMappings.....	PAGE 568
8.	HandlerInterceptor.....	PAGE 588
9.	ViewResolvers In Spring WEB MVC.....	PAGE 596
10.	File Upload in Spring WEB MVC.....	PAGE 611
11.	File Downloading in Spring WEB MVC.....	PAGE 617
12.	Internationalization[I18N].....	PAGE 621
13.	SPRING MVC with Tiles.....	PAGE 631

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring WEB MVC Module

Introduction:

The main intention of Spring WEB MVC module is to prepare web applications with MVC design patterns.

Q) To prepare web applications we have already Struts1 Framework then what is the requirement to use Spring Web MVC Module?

Ans:

1. Struts is Web Framework, it able to provide very good environment to prepare and execute web applications.

Spring Framework is an application Framework, it able to provide very good environment to prepare any type of application like Standalone applications, web applications, Distributed applications,.....

2. Struts framework is mainly MVC based frame work, it able to use only MVC and its corelated design pattern.

In Spring framework, only Web Module is able to use MVC Design Pattern.

3. Struts Framework is heavy weight Framework.

Spring Framework is light weight Framework.

4. Struts framework is able to provide tightly coupled design for its applications.

Spring framework is able to provide loosly coupled design.

5. Struts is not providing clear seperatin between Controller layer, Beans Model and View part.

Spring Framework is able to provide clear seperation between controller layer, Model and View part.

6. Struts is more API dependent, it is very difficult to perform debugging and Testing.

Spring is less API dependent, it is very simple perform debugging and testing.

7. Struts is not providing very good environment to integrate other technology applications like JDBC, Hibernate, EJBs,....

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring is providing very good environment to integrate other technology applications like JDBC, Hibernate,....

8. Struts is able to use only HTML, JSP ,... basic view related tech to prepare View part.

Spring is able to provide very good environment to use view related tech like HTML, JSP, velocity, Freemarker,.....

9. Struts is not having AOP implementations to provide loosely coupled design.

Spring is supporting AOP implementations to provide loosely coupled design.

10. Struts is not layered/Modularized Framework.

Spring is layered/Modularization Framework.

11. Struts is said to be invasive. In Struts we used to extend Action Classes and ActionForm classes. It forces the programmer that, the programmer class must extend from the base class provided by Struts API.

Spring Framework is said to be a non-invasive means it doesn't force a programmer to extend or implement their class from any predefined class or interface given by Spring API.

12. Struts framework is able to provide very good Tag library to prepare view part.

Spring framework is not providing good tag library to prepare view part.

13. Spring is providing very good Transaction management and Messaging support for its applications.

Struts is not providing support for Transaction Management and Messaging Support.

Q) To prepare MVC Based Web Applications we have already JSF[Java Server Faces] then what is the requirement to use Spring Web MVC Framework?

Ans:

1. JSF is web framework, it able to provide environment to prepare web applications only.

Spring Framework is an application Framework, it able to provide very good environment to prepare any type of application like Standalone applications, web applications, Distributed applications,.....

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. JSF is Component based Framework, for every view , JSF will create a Component Tree inorder to manage data. [in order to manage the data form.html data.](#)

Spring Framework is not Component based Framework, it will use Beans to manage Form data.

3. JSF is View layered Framework, It will focus on View Layer.

Spring is not View Layered Framework, it will focus on all layers of tge Enterprise Applications.

[JSF is having very good type convertors, data validations and Rendering mechanisms.](#)

4. JSF is having very good Convertors , Validations and Rendering Mechanisms.

Spring is not having good Convertors, validations and Renderring Mechanisms.

5. JSF is having inbuilt AJAX support.

Spring is not having inbuilt Ajax support.

6. JSF is not having Middleware Services support like Transactions, Messaging,

Spring is having very good support for Transactions, Messaging,....

7. JSF is not having any integration environment to integrate other technologies applications like JDBC, EJBs, RMI,....

Spring is providing very good environment to integrate other technologies applications like JDBC, EJBs, Hibernate, Struts,.....

8. JSF is not having very good Security implementations.

Spring is having a seperate Security module to provide security.

Spring Web MVC Features:

1. Spring Web MVC contains no of components like controller, validator, command object, form object, model object, DispatcherServlet, handler mapping, view resolver, and so on , Each and every component has its own roles and responsibilities.
2. Spring framework allows Powerful and straightforward configuration of both framework and application classes as JavaBeans. This configuration capability includes easy navidation across contexts, such as from web controllers to business objects and validators.
3. Spring Framework is allowing Command and Form objects to reuse Business code instead of extending Frameworks provided API libraries.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



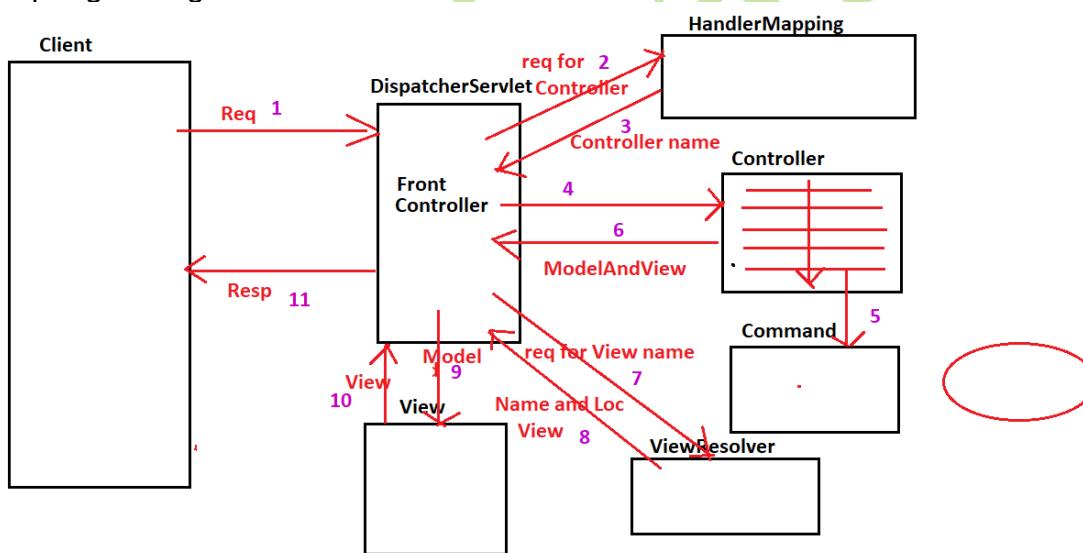
BY NAGOOR BABU

4. Spring Web MVC is providing very good Data Binding Mechanisms, Data Validations ... for web applications.
5. Spring Web MVC is providing Customizable handler mapping and view resolution.
6. Spring Web MVC module is more Flexible model transfer. Model transfer with a name/value Map supports easy integration with any view technology.
7. Spring Web MVC provides customizable locale and theme resolution, support for JSPs with or without Spring tag library, support for JSTL, support for Velocity without the need for extra bridges, and so on.
8. Spring Web MVC allows all JSP supported Tag Libraries like JSTL tag libraries,...

Spring Web MVC Components:

To prepare Spring Web MVC Applications we have to use the following Components.

1. View
2. web.xml
3. DispatcherServlet
4. HandlerMapping
5. Controller Component
6. Command Class
7. View Resolver
8. Spring Configuration File



From the above diagram,

1. Submitting request from Client to Server Side web application, where DispatcherServlet will receive request.
2. DispatcherServlet will interact with HandlerMapping to get Controller name and location.
3. HandlerMapping will return the name and location of the Controller class.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4. DispatcherServlet will recognize the name and location of the Controller class and perform loading and instantiation of the Controller class.
5. Depending on the Controller, Form data will be stored in Command Object and that Command object data will be used in Controller class.
6. After executing Business logic, Controller class will return ModelAndView object, which contains logical name of the view in View object and Model object is a Map contains Model Objects in order to use that data in View part.
7. DispatcherServlet will interact with ViewResolver in order to get the name and location of the View page.
8. ViewResolver will return View object with the name and location of the View page.
9. DispatcherServlet will interact with View part with Model part in order to prepare response.
10. View part will prepare response and submitting that response to DispatcherServlet.
11. DispatcherServlet will generate the required response to Client.

1. View :

Q) What is the requirement to use View part in Web applications?

Ans:

1. It will improve Look And Feel to the web applications.
2. It will provide entry point for the users in order to interact with applications.
3. It will provide very good environment to take data from users in order to submit data to the server side application.
4. It is able to provide very good environment to perform Client Side data validations by using Java Script functions.
5. It allows to specify different types of requests like GET, POST, HEAD,... to the web applications.

There are two types of View parts existed.

1. Informational View:

- It is able to display messages to the users, status of the server side actions.
- It will not include forms, it will not take data from Users, simply it will display data to the users.

2. Form Based View:

- It will include forms to take data from users and to submit that data to the Users.

To prepare View part, we will use View based Tech like AWT, SWING, JAVA FX, html, Jsp, velocity, freemarker,.....

Spring Framework is able to allow all the advanced view tech ,but, at basic level, We will use JSP tech with Spring provided tag library.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. web.xml[Deployment Descriptor]

web.xml is deployment descriptor, it will provide metadata about the web application which is required by the container inorder to identify the server side components and inorder to execute server side components.

In web applicatins, web.xml file is responsible for

1. welcome files configuration
2. Context parameters conf.
3. Servlets Conf.
4. Filters Conf.
5. Listeners Conf.
6. Session time out conf.
7. Error Pages Conf.
8. Tag Libraries conf.
9. Security conf....

In Spring web MVC applicatins , web.xml file requirement is to configure Font Controller that is DispatcherServlet inorder to activate FrontController by Web containers.

In Spring Framework, DispatcherServlet is FrontController provided by Spring Framework in the form of "org.springframework.web.servlet.DispatcherServlet".

In DispatcherServlet conf, we will provide the following configuration details:

a)DispatcherServlet class conf.

1. Logical name of the Dispatcher Servlet conf.
2. DispatcherServlet class conf.
3. Initialization Parameters conf.
4. Load on Startup Configuration.

b)URL Mapping Def.

Where DispatcherServlet must have logical name , on the basis of this logical name only we will prepare Spring configuration file name, that is, logical_Name-servlet.xml

EX: <servlet-name>dispatcherServlet</servlet-name>

Note: in the above case, Spring configuration file name must be dispatcherServlet-Servlet.xml

Where DispatcherServlet class must be configured with its fully qualified name inorder to activate FrontController.

EX: <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where Initialization Parameters are required to create Framework related objects like Handlermapping, HandlerAdapter, ViewResolver,.....

Note: The default name and location of Spring configuration file is under WEB-INF folder with the name "Servlet_Logical_Name-servlet.xml", but, if we want to change this name and location then we must use "contextConfigLocation" initialization parameter in DispatcherServlet configuration.

EX:

1. `<init-param>`
2. `<param-name>contextConfigLocation</param-name>`
3. `<param-value>/WEB-INF/configd/myconfig.xml</param-value>`
4. `</init-param>`

Where load-on-startup configuration is required inorder to perform DispatcherServlet loading, instantiation and initialization at the time of server startup, where DispatcherServlet innitiation is required to load the complete Spring web MVC framework.

EX: `<load-on-startup>1</load-on-startup>`

Where URL pattern definition is required for each and every Servlet in web applications. IN general, in web applications, we will define URL patterns for the Servlets in the following three approaches.

1. Exact Match Method
2. Directory Match Method
3. Extension Match Method

From the above Url pattern definitions, DispatcherServlet required to use either Directory Match method[/*] or Extension Match Method[*.do] inorder to trap all the requests from Clients to DispatcherServlet.

web.xml

1. `<web-app>`
2. `<servlet>`
3. `<servlet-name>dispatcherServlet</servlet-name>`
4. `<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>`
5. `<load-on-startup>1</load-on-startup>`
6. `</servlet>`
7. `<servlet-mapping>`
8. `<servlet-name>dispatcherServlet</servlet-name>`
9. `<url-pattern>*.do</url-pattern> or`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

10. `<url-pattern>/<url-pattern>`
11. `</servlet-mapping>`
12. `</web-app>`

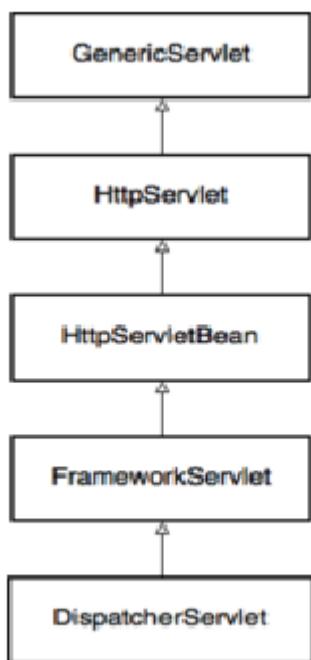
3. DispatcherServlet:

The main intention of the Front Controller in web applications is ,

1. Getting request from clients
2. Identifying the respective Model component or Business component which includes business logic.
3. Executing Business Logic which is existed in Business component.
4. Identifying the respective View part inorder to generate response.
5. Forwarding request to view part inorder to generate dynamic response.

In Spring web mvc module , DispatcherServlet is acting as the Front Controller provided by Spring Framework in the form of "org.springframework.web.servlet.DispatcherServlet".

Spring Framework has provided DispatcherServlet with the following Strucuter.



- Where `HttpServletBean` is the first Spring-aware class in the hierarchy. It injects the bean's properties using the servlet `init-param` values received from the `web.xml` or from `WebApplicationInitializer`.
- Where `FrameworkServlet` integrates the Servlet functionality with a web application context, implementing the `ApplicationContextAware` interface. But it is also able to create a web application context on its own.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

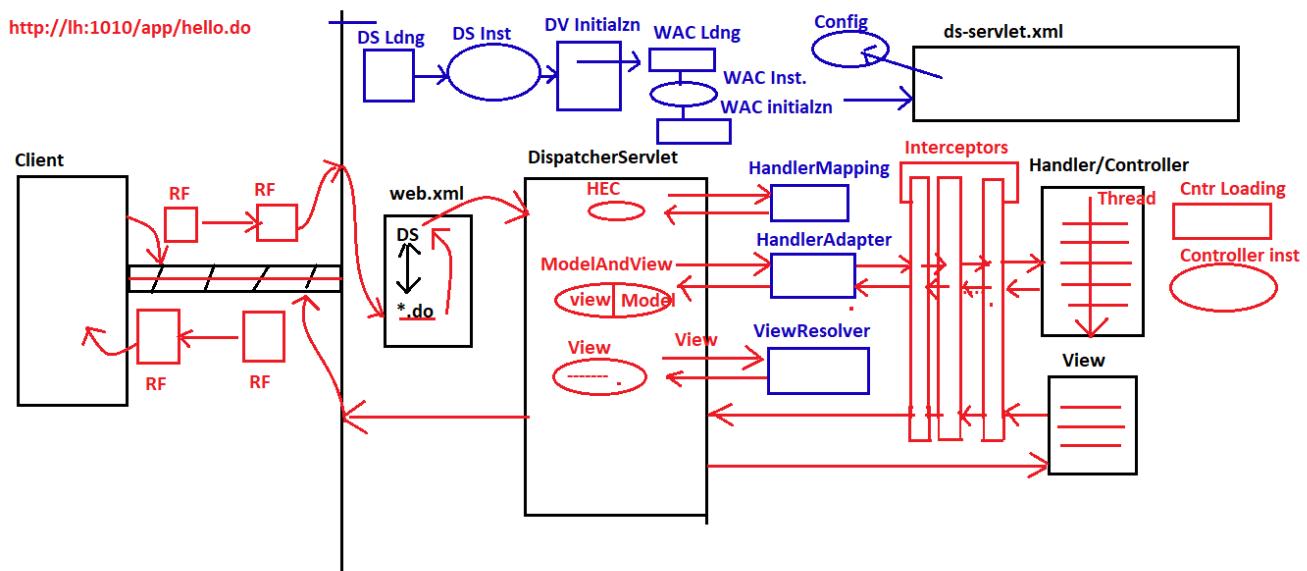
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- Where DispatcherServlet will perform the following actions in Spring web MVC applications in order to process the requests.



From the above diagram,

1. When we start server, container will perform the following actions.
 - a) Container will recognize all the web applications which are existed under webapps folder.
 - b) Container will deploy all the web applications under server space.
 - c) Container will create a separate ServletContext object for each and every web application which we deployed.
2. While deploying web application, container will perform the following actions,
 - a) Container will recognize web.xml file under WEB-INF folder.
 - b) Container will perform web.xml file loading, parsing and reading the content from web.xml file.
 - c) While parsing web.xml file, Container will recognize load-on-startup configuration under DispatcherServlet configuration.
 - d) With the load-on-startup configuration, Container will search for DispatcherServlet under classes folder, if it is not existed then container will search for DispatcherServlet under web application lib folder.
 - e) When DispatcherServlet identifies then container will perform DispatcherServlet loading, instantiation and initialization at the time of server startup or at the time of web application deployment.
3. As part of DispatcherServlet initialization, DispatcherServlet will perform the following actions.
 - a) DispatcherServlet will recognize WebApplicationContext implementation class and perform WebApplicationContext loading, instantiation and initialization.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- b) As part of WebApplicationContext initialization, WebApplicationContext will recognize Struts configuration file under WEB-INF folder with the name [Servlet-Name]-servlet.xml and WebApplicationContext will perform Spring configuration file loading, parsing and reading the content into Configuration object.
 - c) After getting all the configuration details from spring configuration file into Configuration object, WebApplicationContext container will create bean objects which we configured under spring configuration file.
 - d) After getting all bean objects, WebApplicationContext will create Framework objects like HandlerMapping, HandlerAdapter, ViewResolver, ThemeResolver, LocaleResolver,..... as part of WebApplicationContext initialization.
4. When we submit request from client to Server then Protocol will take request and perform the following actions.
- a) Protocol will establish connection between client and server on the basis of the Server IP Address and Port number.
 - b) **Protocol will create request format contains Header part and Body part, where Header part is able to manage Clients metadata and Body part is able to manage Request parameters data which are provided by the user at client browser.**
 - c) Protocol will carry request format to Server, where Server will forward request to web container.
5. When request is coming to the container then Web Container will take URL pattern value from request format and container is trying to compare the URL pattern of the DispatcherServlet which we configured in web.xml file, where if the url pattern is matched with DispatcherServlet URL pattern then container will forward request to DispatcherServlet.
6. When request is coming to the DispatcherServlet then DispatcherServlet will perform the following actions
- ❖ DispatcherServlet will keep all the Framework objects like HandlerMapping, HandlerAdapter, ViewResolver, ThemeResolver, LocaleResolver,..... in request scope in order to make available to the Handler and View part.
 - ❖ DispatcherServlet will interact with HandlerMapping object to get Handler or Controller object by executing getHandler() method.
 - ❖ When DispatcherServlet call getHandler() on HandlerMapping object , HandlerMapping will identify the respective Handler object and its configured interceptors then Handler Mapping will arrange all the interceptors before Handler and return HandlerExecutionChain object which contains Handler object and a set of interceptors.
- Note:** Interceptors are like Filters in servlets, which are used to perform pre-processing and post-processing activities for any web resource.
- ❖ After getting HandlerExecutionChain object, DispatcherServlet will use HandlerAdapter object to access Handler by using handle() method, where HandlerAdapter will execute all

CONTACT US:Mobile: +91- **8885 25 26 27** +91- **7207 21 24 27/28**US NUM: **4433326786**Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

the interceptors inorder to perform pre-processing activities by calling **preHandle()** method.

- ❖ After executing all the interceptors, HandlerAdapter will access business logic which is included in Handler or Controller component.
- ❖ By the execution of Business logic in Handler class, Handler will return ModelAndView object to HandlerAdapter by executing all the interceptors in reverse inorder to provide post-processing activities by executing **postHandle()** method, where HandlerAdapter will send the received ModelAndView object to DispatcherServlet.

Note: Where ModelAndView object contains View object and Model object, where View object contains view name inorder to identify view part and Model object contains no of Objects to provide data to the View part.

- ❖ After getting ModelAndView object, DispatcherServlet will interact with ViewResolver inorder to identify view JSP page url on the basis of View name.
- ❖ When View is identified in web application then DispatcherServlet will execute the corresponding view page by getting data from Model part and DispatcherServlet will prepare Response for the client.
- ❖ When DispatcherServlet prepares response , DispatcherServlet will send response to client, where Protocol will take that response and protocol will perform the following actions.
 - a) Protocol will prepare response format contains Header part and Body part, where Header part is able to manage response headers data like response size, type of response,..... and Body part is able to manage the actual dynamic response.
 - b) Protocol will carry response format to client, where client browser will display the generated response.
 - c) When response is generated at client browser , protocol will destroy the connection which is established between client and Server.
 - d) When we shutdown the server or when we undeploy the web application then container will perform DispatcherServlet deinstantiation, with this, all the Spring Framework objects like HandlerMapping, HandlerAdapter, viewResolver,..... are destroyed.

4. HandlerMapping:

The main intention of HandlerMapping is to map incoming request to the Handler class that can handle requests.

When request is coming to DispatcherServlet, DispatcherServlet will forward request to HandlerMapping , where HandlerMapping will identify the respective Handler class and its associated interceptors then HandlerMapping will create HandlerExecutionChain with Handler class object and interceptors stack and return HandlerExecutionChain to DispatcherServlet.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring Framework has provided the following HandlerMappings inorder to map requests to Handler classes.

- i. BeanNameUrlHandlerMapping
- ii. SimpleUrlHandlerMapping
- iii. ControllerClassNameHandlerMapping
- iv. CommonsPathMapHandlerMapping

Note: IN Spring web MVC applications, if we want to use the HandlerMapping classes then we must configure them in Spring configuration File.

i. BeanNameUrlHandlerMapping

It is default HandlerMapping in Spring applications, it will be used by Spring framework when no HandlerMapping is configured in Spring configuration file.

This HandlerMapping map URLs to beans with names that start with a slash ("/"), similar to how Struts maps URLs to action names.

Spring Framework has provided this Handler mapping like “`org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping`”.

EX:

```
1. <beans>
2. -----
3.   <bean name="/welcome.htm" class="com.durgasoft.controller.WelcomeController"/>
4.   <bean name="/hello.htm" class="com.durgasoft.controller.HelloController"/>
5.   <bean name="handlerMapping" class="org.springframework.web.servlet.handler.Bean
NameUrlHandlerMapping"/>
6. -----
7. </beans>
```

- ❖ In the above configuration, if we submit request with the url <http://localhost:1010/app/welcome.htm> then BeanUrlHandlermapping will forward request to WelcomeController .
- ❖ In the above configuration, if we submit request with the url <http://localhost:1010/app/hello.htm> then BeanUrlHandlermapping will forward request to HelloController .

ii. SimpleUrlHandlerMapping

This HandlerMapping is able to map Handler or Controller classes by matching URL path with the property key of the properties .

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring Framework has provided this HandlerMapping like
“org.springframework.web.servlet.handler.SimpleUrlHandlermapping”.

EX:

```

1. <beans>
2. -----
3. <bean name="addEmp" class="com.durgasoft.controller.AddEmployeeController"/>
4. <bean name="searchEmp" class="com.durgasoft.controller.SearchEmployeeController"/>
5. <bean name="deleteEmp" class="com.durgasoft.controller.DeleteEmployeeController"/>
6. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.Simple
   UrlHandlermapping">
7.   <property name="mappings">
8.     <props>
9.       <prop key="addEmployee.htm">addEmp</prop>
10.      <prop key="searchEmployee.htm">searchEmp</prop>
11.      <prop key="deleteEmployee.htm">deleteEmp</prop>
12.    </props>
13.  </property>
14. </bean>
15. </beans>
```

- ❖ If we submit request like <http://localhost:1010/app/addEmployee.htm> then SimpleUrlHandlerMapping will forward request to AddEmployeeController class.
- ❖ If we submit request like <http://localhost:1010/app/searchEmployee.htm> then SimpleUrlHandlerMapping will forward request to SearchEmployeeController class.
- ❖ If we submit request like <http://localhost:1010/app/deleteEmployee.htm> then SimpleUrlHandlerMapping will forward request to DeleteEmployeeController class.

iii. ControllerClassNameHandlerMapping

This HandlerMapping follows a simple convention for generating URL path mappings from the class names of registered Controller beans as well as @Controller annotated beans. For simple Controller implementations (those that handle a single request type), the convention is to take the short name of the Class, remove the 'Controller' suffix if it exists and return the remaining text, lower-cased, as the mapping, with a leading /. For example:

Spring Framework has provided this HandlerMapping like
“org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping”.

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

**EX:**

```
1. <beans>
2. -----
3. <bean id="welcome" class="com.durgasoft.controller.WelcomeController"/>
4. <bean name="hello" class="com.durgasoft.controller.HelloController"/>
5. <bean name="handlerMapping" class="org.springframework.web.servlet.mvc.support.C
ontrollerClassNameHandlerMapping"/>
6.
7.
8. </beans>
```

- ❖ If we submit request like <http://localhost:1010/app/welcome>* then BeanClassNameHandlerMapping will access WelcomeController class.
- ❖ If we submit request like <http://localhost:1010/app/hello>* then BeanClassNameHandlerMapping

iv. CommonsPathMapHandlerMapping

The org.springframework.web.servlet.handler.CommonsPathMapHandlerMapping is one of the implementation of HandlerMapping interface.

The org.springframework.web.servlet.handler.CommonsPathMapHandlerMapping is designed to recognize Commons attributes meta data attributes of type PathMap defined in the application controller and automatically wires them in to the current DispatcherServlet's Web-application context.

To use this HandlerMapping the controller class must have a class level metadata of the form @org.springframework.web.servlet.handler.commonattributes.PathMap("/mypath.spring"). We can configure multiple path maps for a single controller.

EX:

```
1. @org.springframework.web.servlet.handler.commonattributes.PathMap(
2.           "/mypath.spring")
3.
4. public class FirstController implements Controller {
5.
6.     public ModelAndView handleRequest(HttpServletRequest request,
7.             HttpServletResponse response) throws Exception {
8.         System.out.println("we are in handleRequest()");
9.         ModelAndView mav=null;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

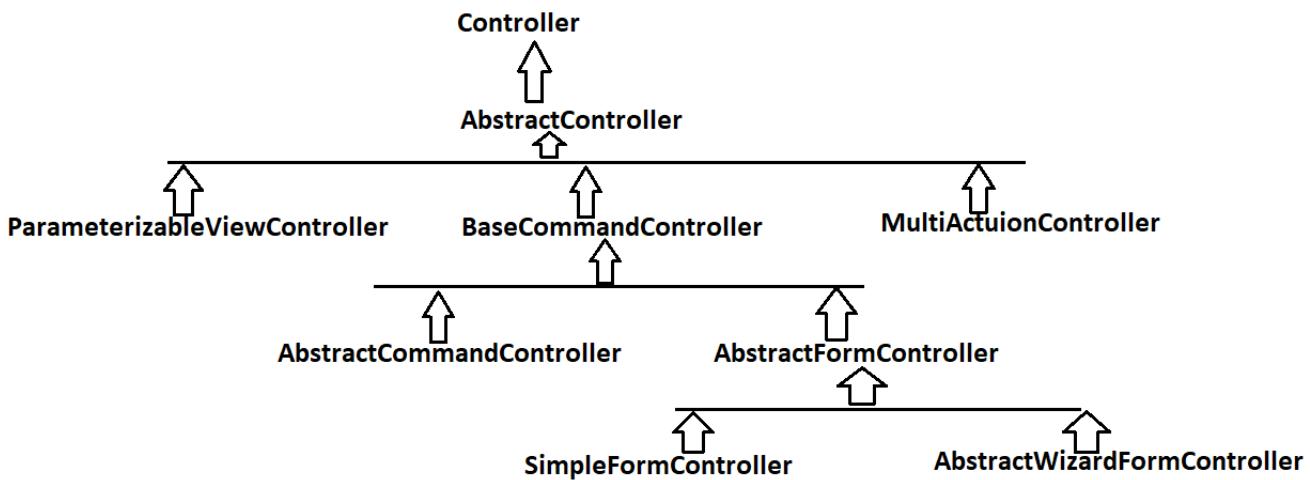


BY NAGOOR BABU

```
10. return mav;
11.}
12.
```

5. Controller Component:

The main intention of Controller in Spring web MVC is to manage application logic or to have controller logic to manage Service layer provided business methods access.
To prepare Controller classes in Spring web MVC applications Spring framework has provided the following predefined Library.



All the above Controller classes are the implementation of `org.springframework.web.servlet.mvc.Controller` interface.

`org.springframework.web.servlet.mvc.Controller` interface has provided the following method to include business logic or controller component logic.

`Public ModelAndView handleRequest(HttpServletRequest req, HttpServletResponse resp) throws Exception`

In Spring web MVC application we can prepare Controller class either by implementing Controller interface or by extending the above Controller classes.

Ex:

1. Public class LoginController implements Controller{
2. Public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception{
3. -----
4. return new ModelAndView("success");
5. }

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where ModelAndView is an object that holds both the model and view. The handler returns the ModelAndView object and DispatcherServlet resolves the view using View Resolvers and View. The View is an object which contains view name in the form of the String and model is a map to add multiple objects.

To create ModelAndView object we have to use the following constructors.

1. Public ModelAndView()
 2. Public ModelAndView(String view_Name)
 3. Public ModelAndView(String view_Name, Map model)
 4. Public ModelAndView(String view_Name, HttpStatus status)
 5. Public ModelAndView(String view_Name, String model_Name, Object model)
-
-

Note:If we want to use Controller classes in Spring web MVC applications then we have to configure in Spring configuration File.

6. Command Class:

The main intention of Command class is to store form data which is submitted by the client when we use FormCommandControllers. It is like ActionForm component in Struts framework.

EX:

1. **public class** User{
 2. **private** String uname;
 3. **private** String upwd;
 4. setXXX() and getXXX()
 5. }
- It is able to take View object from DispatcherServlet and it able to resolve which JSP page and its URL is the respective page and it will return the complete URL of the JSP page to DispatcherServlet.

7. View Resolver:

In Spring MVC, view resolvers enable you to render models in a browser without tying you to a specific view technology like JSP, Velocity, XML...etc.

There are two interfaces that are important to the way Spring handles views are **ViewResolver** and **View**. The ViewResolver provides a mapping between view names and actual views. The View interface addresses the preparation of the request and hands the request over to one of the view technologies.

Spring Framework has provided the following ViewResolvers

- ❖ **ViewResolver**-View Resolvers get a org.springframework.web.servlet.View based on a view name.
- ❖ **ContentNegotiatingViewResolver**-This is an implementation of a view resolver based on the request file name or Accept header. This class delegates view resolution to other view resolvers that are configured. The class uses the MediaType from the request to determine a view. The class first determines the MediaType, then asks each ViewResolver to return a

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

view. The class then compares the all returned views' content type and the one that is most compatible with the request MediaType is chosen.

- ❖ **BeanNameViewResolver**-This resolver implementation resolves a view name as a bean name registered in the application context. In other words, the view is registered as a bean and this resolver gets the name of that bean
- ❖ **UrlBasedViewResolver**-This directly resolves a view name to a URL without any explicit mapping. The view names can be the URL themselves or a prefix or suffix can be added to get the URL from the view name. To redirect a URL use the 'redirect:' prefix. To forward a URL use the 'forward:' prefix.
- ❖ **InternalResourceViewResolver**-This is a subclass of UrlBasedViewResolver and supports an InternalResourceView. An InternalResourceView is a wrapper for JSP or some other resource that reside in the same web application. It exposes the model objects as request attributes. If JSTL API is present then the resolver resolves the view name to a JSTLView. An InternalResourceViewResolver needs to be last, if a chain of view resolvers is used, since this resolves the view name irrespective of whether the actual resource is present or not.
- ❖ **FreeMarkerViewResolver**-This is a subclass of UrlBasedViewResolver that supports FreeMarkerView and its subclasses. The View Class generated can be specified by the 'viewClass' property.
- ❖ **VelocityViewResolver**-This is a subclass of UrlBasedViewResolver that supports VelocityView and its subclasses.
- ❖ **JasperReportsViewResolver**-Supports AbstractJasperReportsView by resolving the view name to the URL of the report file.
- ❖ **XsltViewResolver**- Supports XsltView by resolving the view name to the URL of the XSLT Stylesheet.

IN spring Web MVC applications we have to configure view Resolver in Spring configuration file with the name "viewResolver" id name.

Note: We must configure These View Resolvers in spring Configuration File.

EX: ModeAndView ---->View=success

 /WEB-INF/success.jsp

1. `<bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">`
2. `<property name="prefix" value="WEB-INF"/>`
3. `<property name="suffix" value=".jsp"/>`
4. `</bean>`



8. Spring Configuration File.

In Spring web MVC application, Spring configuration file is recognized by WebApplicationContext container with the name [DS_Logical_Name]-servlet.xml under WEB-INF folder.

The main intention of Spring configuration file in Spring Web MVC application is to provide the configuration details like,

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. Controller classes and their dependencies configuration
2. HandlerMapping configuration.
3. ViewResolver Configuration

EX:

```
1. <beans>
2. -----
3.   <bean name="/welcome.htm" class="com.durgasoft.controller.WelcomeController"/>
4.   <bean name="/hello.htm" class="com.durgasoft.controller.HelloController"/>
5.   <bean name="handlerMapping" class="org.springframework.web.servlet.handler.Bean
NameUrlHandlerMapping"/>
6.   <bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceVi
ewResolver">
7.     <property name="prefix" value="WEB-INF"/>
8.     <property name="suffix" value=".jsp"/>
9.   </bean>
10. -----
11. </beans>
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Steps to prepare Spring web MVC Application

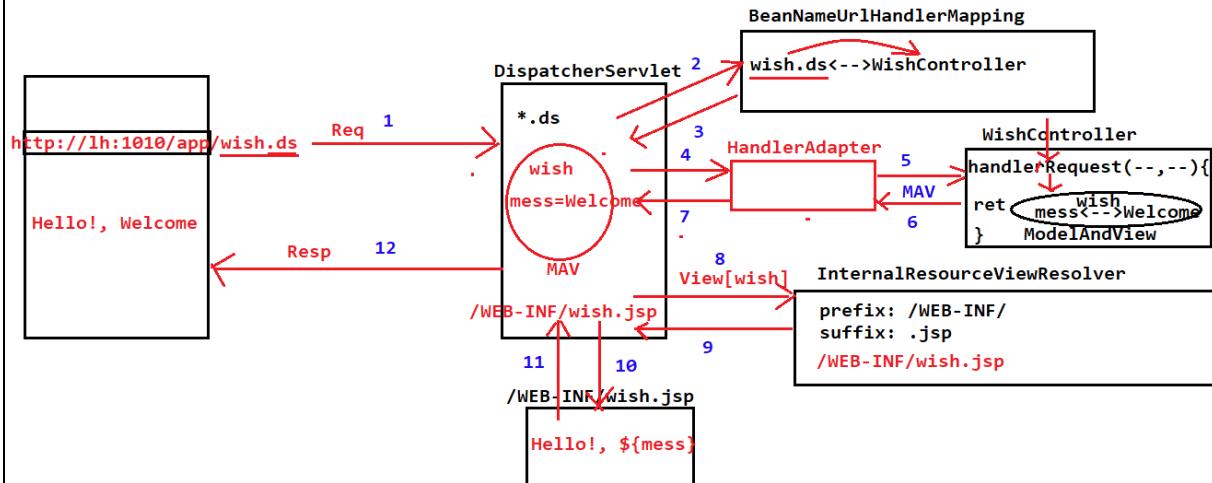
Steps to prepare Spring web MVC Application:

1. Create Dynamic web project with all spring JAR files in lib folder.
2. Prepare user forms as per the requirement.
3. Prepare Controller component and Command classes as per the requirement.
4. Prepare Spring configuration File
5. Create or Edit web.xml file with DispatcherServlet Configuration File.

Examples:

app1

```
app1
|---index.jsp
|---src
|   |----com.durgasoft.controllers.WishController.java
|---WEB-INF
|   |----web.xml
|   |----ds-servlet.xml
|   |----wish.jsp
|   |----classes
|       |----com.durgasoft.controllers.WishController.class
```



wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.    pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10.<body>
11.<h1>
12.Hello User!, ${message} .
13.</h1>
14.</body>
15.</html>
```

WishController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5. import org.springframework.web.servlet.ModelAndView;
6. import org.springframework.web.servlet.mvc.Controller;
7.
8. public class WishController implements Controller {
9.
10.    @Override
11.    public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception {
12.        ModelAndView mav = new ModelAndView("wish","message", "Welcome to Spring WE
13.        B MVC World");
14.        return mav;
15.    }
16.}
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.         xmlns:p="http://www.springframework.org/schema/p"
5.         xmlns:context="http://www.springframework.org/schema/context"
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6.    xsi:schemaLocation="
7.        http://www.springframework.org/schema/beans
8.        http://www.springframework.org/schema/beans/spring-beans.xsd
9.        http://www.springframework.org/schema/context
10.       http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.   <bean name="/wish.ds" class="com.durgasoft.controllers.WishController"/>
13.   <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
14.   <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
15.     <property name="prefix" value="/WEB-INF/"/>
16.     <property name="suffix" value=".jsp"/>
17.   </bean>
18. </beans>
```

web.xml

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
3.    <display-name>app1</display-name>
4.    <welcome-file-list>
5.      <welcome-file>index.html</welcome-file>
6.      <welcome-file>index.htm</welcome-file>
7.      <welcome-file>index.jsp</welcome-file>
8.      <welcome-file>default.html</welcome-file>
9.      <welcome-file>default.htm</welcome-file>
10.     <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>*.ds</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Application-2**index.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <center>
11. <br><br>
12. <h3>
13. <a href="http://localhost:1010/app2/hello">hello</a>
14. <br><br>
15. <a href="http://localhost:1010/app2/welcome">welcome</a>
16. </h3>
17. </center>
18. </body>
19. </html>
```

hello.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h1>
11. Hello User!, This is Spring WEB MVC.
12. </h1>
13. </body>
14. </html>
```

welcome.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.  pageEncoding="ISO-8859-1">%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10.<h1>Welcome User!, This is Spring WEB MVC.</h1>
11.</body>
12.</html>
```

HelloController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
9. public class HelloController implements Controller {
10.
11.     @Override
12.     public ModelAndView handleRequest(HttpServletRequest arg0, HttpServletResponse ar
g1) throws Exception {
13.         ModelAndView mav = new ModelAndView("hello");
14.         return mav;
15.     }
16.
17. }
```

WelcomeController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9. public class WelcomeController implements Controller {
10.
11.   @Override
12.   public ModelAndView handleRequest(HttpServletRequest arg0, HttpServletResponse ar
g1) throws Exception {
13.
14.     ModelAndView mav = new ModelAndView("welcome");
15.     return mav;
16.   }
17. }
```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.   <bean name="/hello" class="com.durgasoft.controllers.HelloController"/>
13.   <bean name="/welcome" class="com.durgasoft.controllers.WelcomeController"/>
14.
15.   <bean name="handlerMapping" class="org.springframework.web.servlet.handler.Bea
nNameUrlHandlerMapping"/>
16.
17.   <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalR
esourceViewResolver">
18.     <property name="prefix" value="/WEB-INF/"/>
19.     <property name="suffix" value=".jsp"/>
20.   </bean>
21. </beans>
```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



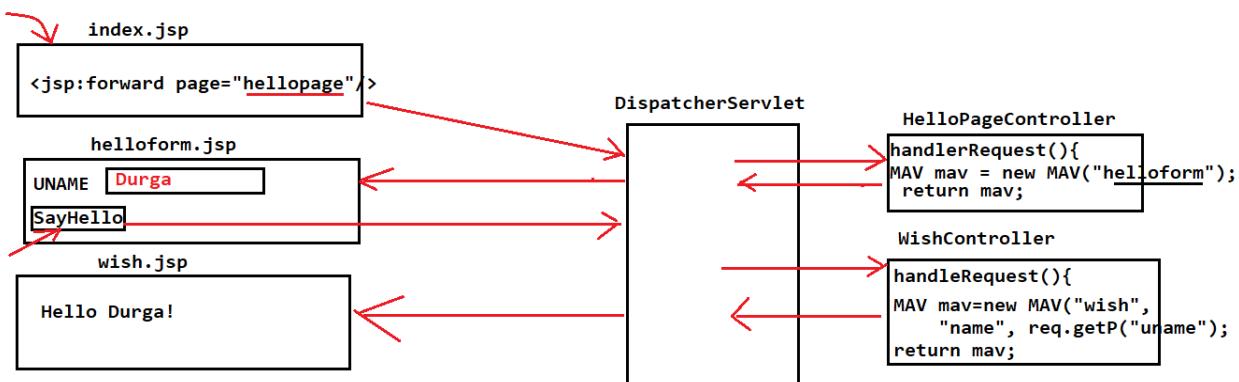
BY NAGOOR BABU

```

org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3. <display-name>app2</display-name>
4. <welcome-file-list>
5. <welcome-file>index.html</welcome-file>
6. <welcome-file>index.htm</welcome-file>
7. <welcome-file>index.jsp</welcome-file>
8. <welcome-file>default.html</welcome-file>
9. <welcome-file>default.htm</welcome-file>
10. <welcome-file>default.jsp</welcome-file>
11. </welcome-file-list>
12. <servlet>
13. <servlet-name>ds</servlet-name>
14. <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15. <load-on-startup>1</load-on-startup>
16. </servlet>
17. <servlet-mapping>
18. <servlet-name>ds</servlet-name>
19. <url-pattern>/</url-pattern>
20. </servlet-mapping>
21. </web-app>

```

Application-3



index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. </head>
9. <body>
10. <jsp:forward page="hellopage"/>
11. </body>
12. </html>
```

helloform.html

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2>Durga Software Solutions</h2>
11. <h3>User Hello Form</h3>
12. <form action="wish">
13. <table>
14. <tr>
15.   <td>User Name</td>
16.   <td><input type="text" name="uname"/></td>
17. </tr>
18. <tr>
19.   <td><input type="submit" value="SayHello"/></td>
20. </tr>
21. </table>
22. </form>
23. </body>
24. </html>
```

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. </head>
10. <body>
11. <h1>
12. Hello ${name}!
13. </h1>
14. </body>
15. </html>
```

HelloPageController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7.
8. import org.springframework.web.servlet.mvc.Controller;
9.
10.
11. public class HelloPageController implements Controller {
12.
13.     @Override
14.     public ModelAndView handleRequest(HttpServletRequest arg0, HttpServletResponse arg1) throws Exception {
15.
16.         ModelAndView mav = new ModelAndView("helloform");
17.         return mav;
18.     }
19.
20. }
```

WishController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
9. public class WishController implements Controller {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10.
11. @Override
12. public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception {
13.     ModelAndView mav = new ModelAndView("wish", "name", request.getParameter("uname"));
14.     return mav;
15. }
16.

```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <bean name="/hellopage" class="com.durgasoft.controllers.HelloPageController"/>
13. <bean name="/wish" class="com.durgasoft.controllers.WishController"/>
14.
15. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
16.
17. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
18.   <property name="prefix" value="/WEB-INF/"/>
19. <?xml version="1.0" encoding="UTF-8"?>
20. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
21.   <display-name>app3</display-name>
22.
23.   <servlet>
24.     <servlet-name>ds</servlet-name>
25.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
26.     <load-on-startup>1</load-on-startup>
27.   </servlet>
28.   <servlet-mapping>
29.     <servlet-name>ds</servlet-name>
30.     <url-pattern>/</url-pattern>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

31. </servlet-mapping>
32. </web-app>
33.
34. </bean>
35.</beans>
```

web.xml


```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app3</display-name>
4.
5.   <servlet>
6.     <servlet-name>ds</servlet-name>
7.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
8.     <load-on-startup>1</load-on-startup>
9.   </servlet>
10.  <servlet-mapping>
11.    <servlet-name>ds</servlet-name>
12.    <url-pattern>/</url-pattern>
13.  </servlet-mapping>
14. </web-app>
```


Application-4:**index.jsp**


```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5.   <head>
6.     <meta charset="ISO-8859-1">
7.     <title>Insert title here</title>
8.   </head>
9.   <body>
10.    <jsp:forward page="loginpage"/>
11.  </body>
12. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

loginform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2>Durga Software Solutions</h2>
11. <h3>User Login Page</h3>
12. <form method="POST" action="login">
13. <table>
14. <tr>
15.   <td>User Name</td>
16.   <td><input type="text" name="uname"/></td>
17. </tr>
18. <tr>
19.   <td>Password</td>
20.   <td><input type="password" name="upwd"/></td>
21. </tr>
22. <tr>
23.   <td><input type="submit" value="Login"/></td>
24. </tr>
25. </table>
26. </form>
27. </body>
28. </html>
```

Success.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2>Durga Software Solutions</h2>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11. <h3>User Login Status</h3>
12. <br><br>
13. <h1> Login Success</h1>
14. </body>
15. </html>
```

failure.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2>Durga Software Solutions</h2>
11. <h3>User Login Status</h3>
12. <br><br>
13. <h1> Login Failure</h1>
14. </body>
15. </html>
```

LoginPageController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
9. public class LoginPageController implements Controller {
10.
11.     @Override
12.     public ModelAndView handleRequest(HttpServletRequest arg0, HttpServletResponse arg1) throws Exception {
13.
14.
15.         return new ModelAndView("loginform");
16.     }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

LoginController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
9. public class LoginController implements Controller {
10.
11.     @Override
12.     public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception {
13.         String uname = request.getParameter("uname");
14.         String upwd = request.getParameter("upwd");
15.         ModelAndView mav = null;
16.         if(uname.equals("durga") && upwd.equals("durga")) {
17.             mav = new ModelAndView("success");
18.         }else {
19.             mav = new ModelAndView("failure");
20.         }
21.         return mav;
22.     }
23. }
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.         xmlns:p="http://www.springframework.org/schema/p"
5.         xmlns:context="http://www.springframework.org/schema/context"
6.         xsi:schemaLocation="
7.             http://www.springframework.org/schema/beans
8.             http://www.springframework.org/schema/beans/spring-beans.xsd
9.             http://www.springframework.org/schema/context
10.            http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.     <bean name="/loginpage" class="com.durgasoft.controllers.LoginPageController"/>
13.     <bean name="/login" class="com.durgasoft.controllers.LoginController"/>
14.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
15. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
16.
17. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
18.   <property name="prefix" value="/WEB-INF/"/>
19.   <property name="suffix" value=".jsp"/>
20. </bean>
21.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app4</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21.</web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Annotations in Spring WEB MVC

Annotations in Spring WEB MVC

Spring Framework has provided annotations support right from its Spring 2.5 version. By using Annotations in Spring Web MVC applications we are able to reduce the following configurations in spring configuration file.

- + We can remove all Controller classes configuration.
- + We can remove HandlerMapping Class configuration.

Spring Web MVC Framework has provided the following Annotations mainly.

1. @Controller
2. @RequestMapping
3. @RequestParam
4. @SessionAttributes

1. @Controller:

Before Spring 2.5, all controller classes must be configured in spring configuration file, But, from Spring 2.5 version, We are able to declare all controller classes with @Controller annotation to tell the Spring container that this class is a controller. In this context, to give an information to the Spring Container about the Controller classes and their locations we have to add the following tag in spring configuration file.

```
<context:component-scan base-package="package name where controller classes are existed" />
```

The above declaration(context:component-scan) tells the Spring container to auto scan the entire package to identify the controller components.

EX:

```
@Controller  
public class LoginController {  
----  
}
```

2. @RequestMapping

@RequestMapping annotation is used for defining the request urls based on the context root and HTTP request methods like GET , POST, HEAD,... for the request.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

This Annotation is used as either Class level annotation or Method level Annotation.

Syntax:

```
@RequestMapping(value="/---", method=---
```

Where value member will take Request URL with / prefix.

Where method member will take the constants like GET, POST, HEAD,... from RequestMethod enum.

EX:

```
@Controller
public class LoginController {
    @RequestMapping(value="/login", method=RequestMethod.POST)
    public ModelAndView checkLogin(HttpServletRequest request, HttpServletResponse
response){
        -----
        return new ModelAndView("success");
    }
}
```

Note: It is not mandatory to return ModelAndView object from controller method, where we can return String also but the parameter to Controller Method must be ModelMap, it will take model attributes which we want to submit to View JSP page.

EX:

```
@Controller
public class HelloController {
    @RequestMapping(value = "/hello", method = RequestMethod.GET)
    public String printHello(ModelMap model) {
        model.addAttribute("message", "Hello Spring MVC Framework!");
        return "hello";
    }
}
```

3. @RequestParam:

@RequestParam annotation will be used in the methods to bind the method parameters to the request parameters. If the request is submitted by the browser, request parameters are passed through the URL query parameters. That can be easily mapped to the methods by annotating the method parameters with @RequestParam.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Syntax:

```
@RequestParam("paramName")
```

EX:

```
@Controller  
public class LoginController {  
    @RequestMapping(value="/login", method=RequestMethod.POST)  
    public ModelAndView checkLogin(@RequestParameter("uname") String uname,  
    @RequestParameter("upwd") String upwd){  
        -----  
        return new ModelAndView("success");  
    }  
}
```

4. @SessionAttributes:

@SessionAttributes is used to define a variable name inorder to keep its Key-Value pair in Session Scope and inorder to use that in multiple pages in the same web application.

Syntax:

```
@SessionAttributes("Param_Name")
```

EX:

```
@Controller  
@SessionAttributes("status")  
public class LoginController {  
    @RequestMapping(value="/login", method=RequestMethod.POST)  
    public ModelAndView checkLogin(@RequestParameter("uname") String uname,  
    @RequestParameter("upwd") String upwd, ModelMap map){  
        -----  
        map.addAttribute("status", "Login SUCCESS");  
        return new ModelAndView("success");  
    }  
}
```

status.jsp

1. <h1>
2. <%
3. String status = session.getAttribute("status");

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
4. out.println(status);
5. %>
6. <h1>
```

Example-5:**index.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="loginpage"/>
11. </body>
12. </html>
```

loginform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2>Durga Software Solutions</h2>
11. <h3>User Login Page</h3>
12. <form method="POST" action="login">
13. <table>
14. <tr>
15.   <td>User Name</td>
16.   <td><input type="text" name="uname"/></td>
17. </tr>
18. <tr>
19.   <td>Password</td>
20.   <td><input type="password" name="upwd"/></td>
```

CONTACT US:**Mobile: +91- 8885 25 26 27**

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
21. </tr>
22. <tr>
23.   <td><input type="submit" value="Login"/></td>
24. </tr>
25. </table>
26. </form>
27. </body>
28. </html>
```

status.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h1>
11. Status : <%= session.getAttribute("status") %>
12. </h1>
13. <h3>
14. <a href="loginpage">|Login Form|</a>
15. </h3>
16. </body>
17. </html>
```

LoginController.java

```
1. package com.durgasoft.controllers;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.ui.ModelMap;
7. import org.springframework.web.bind.annotation.RequestMapping;
8. import org.springframework.web.bind.annotation.RequestParam;
9. import org.springframework.web.bind.annotation.SessionAttributes;
10. import org.springframework.web.servlet.ModelAndView;
11.
12.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

Mail ID: durgasoftonlinetraining@gmail.com +91- 7207 21 24 27/28WEBSITE: www.durgasoftonline.com

US NUM: 4433326786

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

13. @org.springframework.stereotype.Controller
14. @SessionAttributes("status")
15. public class LoginController {
16.
17.     @RequestMapping("/loginpage")
18.     public String loginPage(){
19.         return "loginform";
20.     }
21.
22.
23.     @RequestMapping("/login")
24.     public String login(@RequestParam("uname") String uname, @RequestParam("upwd")
String upwd, ModelMap map)throws Exception {
25.         String status = "";
26.         if(uname.equals("durga") && upwd.equals("durga")) {
27.             map.addAttribute("status", "Login Success");
28.         }else {
29.             map.addAttribute("status", "Login Failure");
30.         }
31.         return "status";
32.     }
33. }
```

ds-servlet.java

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:p="http://www.springframework.org/schema/p"
5.     xmlns:context="http://www.springframework.org/schema/context"
6.     xsi:schemaLocation="
7.         http://www.springframework.org/schema/beans
8.         http://www.springframework.org/schema/beans/spring-beans.xsd
9.         http://www.springframework.org/schema/context
10.        http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <context:component-scan base-package="com.durgasoft.controllers"/>
13. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalReso
urceViewResolver">
14.     <property name="prefix" value="/WEB-INF/"/>
15.     <property name="suffix" value=".jsp"/>
16. </bean>
17. </beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app4</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```



CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring Web MVC Application with MAVEN

Spring Web MVC Application with MAVEN

1. Create Maven Project
2. Select Archetype: web
3. Update pom.xml
4. Update web.xml
5. Provide all View Resources
6. Provide all Java Resources
7. Provide XML files
8. Run web application

index.jsp

```
1. <html>
2. <body>
3. <jsp:forward page="hellopage"/>
4. </body>
5. </html>
```

helloform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red">Durga Software Solutions</h2>
11. <h3 style="color: blue">User Hello Page</h3>
12. <form action="wish">
13. <table>
14. <tr>
15.     <td>User Name</td>
16.     <td><input type="text" name="uname"/></td>
17. </tr>
18. <tr>
19.     <td><input type="submit" value="SayHello"/></td>
20. </tr>
21. </table>
22. </form>
23. </body>
24. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <h1>
12. ${message}
13. </h1>
14. <h3>
15. <a href="hellopage">Hello Page|</a>
16. </h3>
17. </body>
18. </html>
```

HelloController.java

```
1. package com.durgasoft.controllers;
2.
3. import org.springframework.stereotype.Controller;
4. import org.springframework.ui.ModelMap;
5. import org.springframework.web.bind.annotation.RequestMapping;
6. import org.springframework.web.bind.annotation.RequestParam;
7.
8. @Controller
9. public class HelloController {
10.   @RequestMapping("/hellopage")
11.   public String helloPage() {
12.
13.     return "helloform";
14.   }
15.
16.   @RequestMapping("/wish")
17.   public String wish(@RequestParam("uname") String uname, ModelMap map) {
18.     map.addAttribute("message", "Hello " + uname + "!");
19.     return "wish";
20.   }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.  <context:annotation-config/>
12.  <context:component-scan base-package="com.durgasoft.controllers"/>
13.
14.  <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalReso
urceViewResolver">
15.    <property name="prefix" value="/WEB-INF/">
16.    <property name="suffix" value=".jsp"/>
17.  </bean>
18. </beans>
```

web.xml

```
1. <!DOCTYPE web-app PUBLIC
2. "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3. "http://java.sun.com/dtd/web-app_2_3.dtd" >
4.
5. <web-app>
6.  <display-name>Archetype Created Web Application</display-name>
7.  <servlet>
8.    <servlet-name>ds</servlet-name>
9.    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
10.   <load-on-startup>1</load-on-startup>
11.  </servlet>
12.  <servlet-mapping>
13.    <servlet-name>ds</servlet-name>
14.    <url-pattern>/</url-pattern>
15.  </servlet-mapping>
16. </web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

pom.xml

```
1. <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
3.   <modelVersion>4.0.0</modelVersion>
4.   <groupId>com.durgasoft</groupId>
5.   <artifactId>simpleapp</artifactId>
6.   <packaging>war</packaging>
7.   <version>0.0.1-SNAPSHOT</version>
8.   <name>simpleapp Maven Webapp</name>
9.   <url>http://maven.apache.org</url>
10.  <dependencies>
11.    <dependency>
12.      <groupId>junit</groupId>
13.      <artifactId>junit</artifactId>
14.      <version>3.8.1</version>
15.      <scope>test</scope>
16.    </dependency>
17.
18.    <dependency>
19.      <groupId>javax.servlet</groupId>
20.      <artifactId>javax.servlet-api</artifactId>
21.      <version>3.1.0</version>
22.      <scope>provided</scope>
23.    </dependency>
24.
25.    <dependency>
26.      <groupId>org.springframework</groupId>
27.      <artifactId>spring-core</artifactId>
28.      <version>4.3.9.RELEASE</version>
29.    </dependency>
30.
31.
32.    <dependency>
33.      <groupId>org.springframework</groupId>
34.      <artifactId>spring-context</artifactId>
35.      <version>4.3.9.RELEASE</version>
36.    </dependency>
37.
38.
39.    <dependency>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
40. <groupId>org.springframework</groupId>
41. <artifactId>spring-web</artifactId>
42. <version>4.3.9.RELEASE</version>
43. </dependency>
44.
45. <dependency>
46.   <groupId>org.springframework</groupId>
47.   <artifactId>spring-webmvc</artifactId>
48.   <version>4.3.9.RELEASE</version>
49. </dependency>
50.
51. </dependencies>
52. <build>
53.   <finalName>simpleapp</finalName>
54.   <plugins>
55.     <plugin>
56.       <groupId>org.apache.maven.plugins</groupId>
57.       <artifactId>maven-compiler-plugin</artifactId>
58.       <version>3.8.0</version>
59.       <configuration>
60.         <source>1.8</source>
61.         <target>1.8</target>
62.       </configuration>
63.     </plugin>
64.     <plugin>
65.       <groupId>org.apache.tomcat.maven</groupId>
66.       <artifactId>tomcat7-maven-plugin</artifactId>
67.       <version>2.3-SNAPSHOT</version>
68.     </plugin>
69.   </plugins>
70. </build>
71. </project>
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

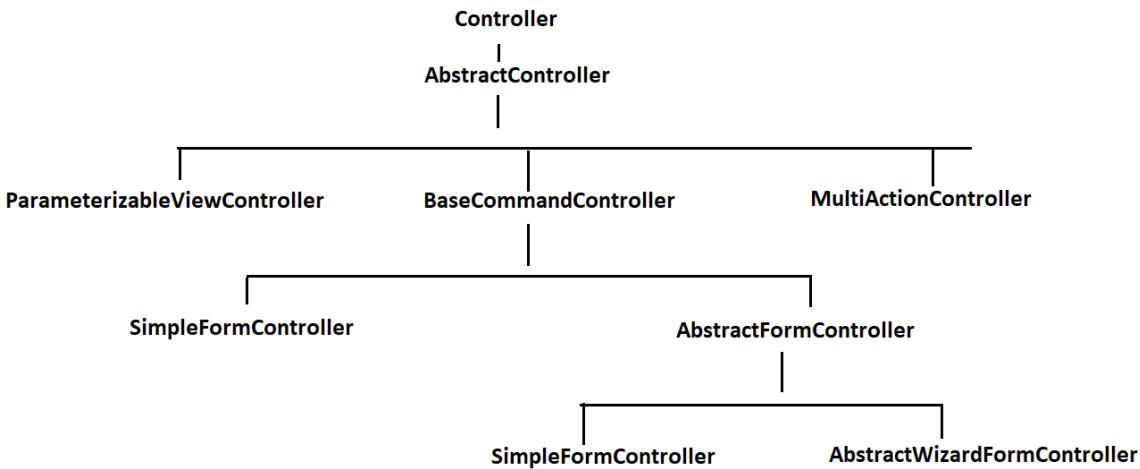


BY NAGOOR BABU

Controller Classes

The main intention of Controller classes is to manage business logic or to include code in order to access the business logic existed with Business components.

To prepare Controller classes, Spring framework has provided a set of predefined classes in "org.springframework.web.servlet.mvc" package which are implemented by Controller interface.



AbstractController:

This Controller Class will be used to prepare Controller classes where no form data submission is required, but, it will display dynamic content through web pages.

AbstractController class is able to have the following method to have Business logic or the code to access business components.

`public ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response) throws Exception`

In general, we will use this Controller class in the following situations.

- When we click on "view profile" link in facebook account, generating profile data
- When we click on "Get all Jobs" link in JOB Portals

Note: In all the cases, no form submission is going on, but, dynamic content will be included in response.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**index.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="wish"/>
11. </body>
12. </html>
```

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <h1 style="color:red">Hello User ${message} </h1>
12. </body>
13. </html>
```

WishController.java

```
1. package com.durgasoft.controller;
2.
3.
4. import java.time.LocalTime;
5.
6. import javax.servlet.http.HttpServletRequest;
7. import javax.servlet.http.HttpServletResponse;
8.
9. import org.springframework.web.servlet.ModelAndView;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10. import org.springframework.web.servlet.mvc.AbstractController;
11.
12. public class WishController extends AbstractController {
13.
14.     @Override
15.     protected ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response) throws Exception {
16.         LocalTime time = LocalTime.now();
17.         int hour = time.getHour();
18.         String wish_Message = "";
19.         if(hour<12) {
20.             wish_Message = "Good Morning";
21.         }else if(hour < 17 ) {
22.             wish_Message = "Good After noon";
23.         }else {
24.             wish_Message = "Good Evening";
25.         }
26.         return new ModelAndView("wish", "message", wish_Message);
27.     }
28. }
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:p="http://www.springframework.org/schema/p"
5.     xmlns:context="http://www.springframework.org/schema/context"
6.     xsi:schemaLocation="
7.         http://www.springframework.org/schema/beans
8.         http://www.springframework.org/schema/beans/spring-beans.xsd
9.         http://www.springframework.org/schema/context
10.        http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.    <bean name="/wish" class="com.durgasoft.controller.WishController"/>
13.    <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
14.    <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
15.        <property name="prefix" value="/WEB-INF/"/>
16.        <property name="suffix" value=".jsp"/>
17.    </bean>
18. </beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <servlet>
4.     <servlet-name>ds</servlet-name>
5.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
6.     <load-on-startup>1</load-on-startup>
7.   </servlet>
8.   <servlet-mapping>
9.     <servlet-name>ds</servlet-name>
10.    <url-pattern>/</url-pattern>
11.  </servlet-mapping>
12.
13.</web-app>
```

There are three types of AbstractController classes.

1. ParameterizableViewController
2. MultiActionController
3. BaseCommandController

1. ParameterizableViewController

- Spring framework has provided this controller class in the form of "org.springframework.web.servlet.mvc.ParameterizableViewController".
- The main intention of this controller class is to display web pages without processing request.
- In ParameterizableViewController classes, it is not required to define any user defined controller class, rather, it must be configured in Spring configuration file directly and it will take "viewName" property with the logical name of the view name in order to open JSP page.
- In general, we will use mechanism when we want to get some web pages just by clicking on hyper links.

EX: If we click on "contactUs" and "aboutUs" links in webpages then we are able to get contactUs web page and aboutUs web page.

Example:
index.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="home"/>
11. </body>
12. </html>
```

home.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <h1 align="center">
12. <a href="contactUs">ContactUs</a>
13. <a href="aboutUs">AboutUs</a>
14. </h1>
15. </body>
16. </html>
```

contactUs.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color:red" align="center">
11. Durga Software Solutions<br>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

12. 202, HMDA, Mitrivanam, Hyd-38
 13. Website: www.durgasoft.com
 14. Mobile: 9246212143, 8885252627
 15. <h2>

 16. <h3>
 17. Home Page
 18. </h3>
 19. </body>
 20. </html>

aboutUs.jsp

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
 2. pageEncoding="ISO-8859-1"%>
 3. <!DOCTYPE html>
 4. <html>
 5. <head>
 6. <meta charset="ISO-8859-1">
 7. <title>Insert title here</title>
 8. </head>
 9. <body>
 10. <h2 style="color:red" align="center">
 11. Durga Software Solutions is an IT Training Center, it has 40 branches through out the world
 .

 12. Durga Software Solutions has started in 2007, Oct, 2nd on the occasion of Gandhi Jayanthi
 .

 13. Durga Software Solutions has started Development Center at Madhapur to provide Software Services.
 14. </h2>

 15. <h3>
 16. Home Page
 17. </h3>
 18. </body>
 19. </html>

HomeController.java

1. package com.durgasoft.controller;
 2.
 3.
 4. import javax.servlet.http.HttpServletRequest;
 5. import javax.servlet.http.HttpServletResponse;
 6.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7. import org.springframework.web.servlet.ModelAndView;
8. import org.springframework.web.servlet.mvc.AbstractController;
9.
10. public class HomeController extends AbstractController {
11.
12.     @Override
13.     protected ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response) throws Exception {
14.         return new ModelAndView("home");
15.     }
16. }
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.         xmlns:p="http://www.springframework.org/schema/p"
5.         xmlns:context="http://www.springframework.org/schema/context"
6.         xsi:schemaLocation="
7.             http://www.springframework.org/schema/beans
8.             http://www.springframework.org/schema/beans/spring-beans.xsd
9.             http://www.springframework.org/schema/context
10.            http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.     <bean name="/home" class="com.durgasoft.controller.HomeController"/>
13.     <bean name="/contactUs" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
14.         <property name="viewName" value="contactUs"/>
15.     </bean>
16.     <bean name="/aboutUs" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
17.         <property name="viewName" value="aboutUs"/>
18.     </bean>
19.     <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
20.     <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
21.         <property name="prefix" value="/WEB-INF/"/>
22.         <property name="suffix" value=".jsp"/>
23.     </bean>
24. </beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

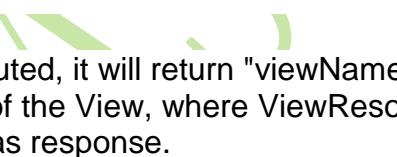
web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <servlet>
4.     <servlet-name>ds</servlet-name>
5.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
6.     <load-on-startup>1</load-on-startup>
7.   </servlet>
8.   <servlet-mapping>
9.     <servlet-name>ds</servlet-name>
10.    <url-pattern>/</url-pattern>
11.  </servlet-mapping>
12. </web-app>
```


Internal Flow:

handleRequestInternal() method will be executed, it will return "viewName" property value as ModelAndView object , that is, logical name of the View, where ViewResolver will resolve the View Page and generate the respective webpage as response.


EX:

```
public class ParameterizableViewController extends AbstractController{
  public ModelAndView handleRequestInternal(HttpServletRequest request,
HttpServletResponse response) throws Exception{
    return new ModelAndView(getViewName());
}
```


MultiActionController:

In general, in Spring WEB MVC applications, we will for each and every form or an URI we will prepare a separate Controller class depending on our requirement.

In the above approach, if we want to provide a particular Entity related actions like save, update, search and delete then we have to prepare separate Controller classes like SaveController, UpdateController, SearchController, DeleteController... it will increase no of controller classes unnecessarily.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In the above context, to reduce no of controller classes, SPring Framework has provided an alternative in the fomr of

"org.springframework.web.servlet.mvc.mviaction.MultiActionController".

The main intention of MultiActionController class is to group related actions into a single controller class.

If we want to use MultiAction controller class in Spring WEB MVC applications then we have to declare an user defined class as sub class to "MultiActionController" and we have to define action methods with the same incoming URI names.

```
public (ModelAndView | Map | String | void) actionPerformed(HttpServletRequest request,
HttpServletResponse response);
```

Ex:

```
1. public class StudentController extends MultiActionController{
2. public ModelAndView save(HttpServletRequest req, HttpServletResponse res) throws Exception{
3.   ----
4. }
5. public ModelAndView search(HttpServletRequest req, HttpServletResponse res) throws Exception{
6.   ----
7. }
8. public ModelAndView update(HttpServletRequest req, HttpServletResponse res) throws Exception{
9.   ----
10.}
11. public ModelAndView delete(HttpServletRequest req, HttpServletResponse res) throws Exception{
12.   ----
13.}
14.}
15.
16. To trap all request to the MultiActionController class and to execute action methods we have to use '/' as "name" attribute value in "bean" configuration in Spring Configuration File.
17.
18. <bean name="/" class="com.durgasoft.controller.StudentController"/>
19.
20. In the above context, if we submit "http://lh:1010/app/save" then save() method in StudentAction class will be executed, similarly, search(), update(), delete().. methods will be executed as per the URI values from clients like search, update, delete...
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

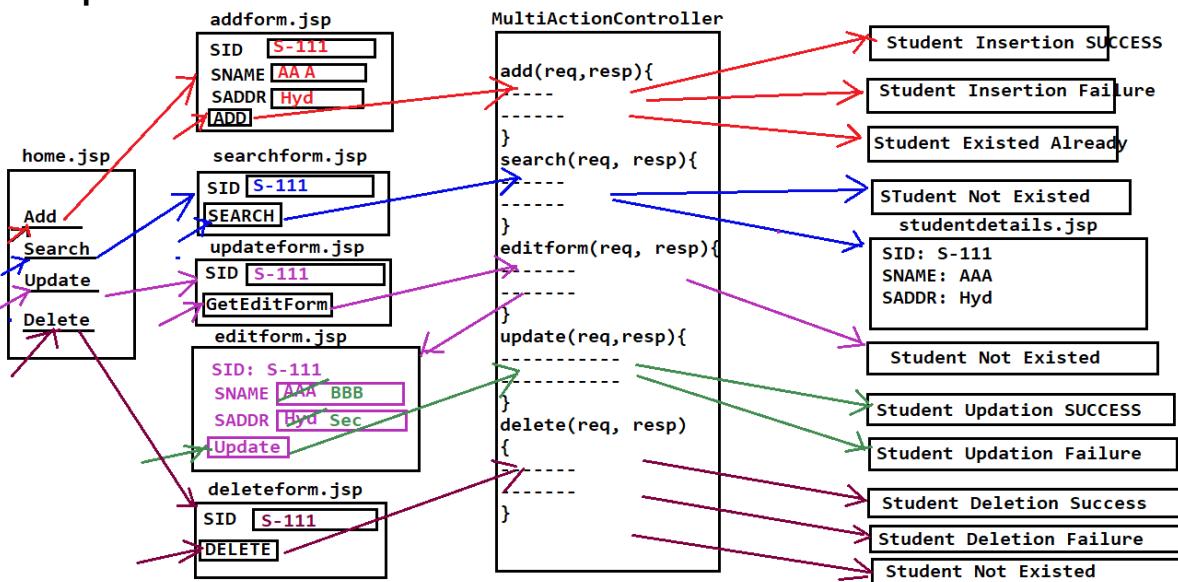
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:



index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="home"/>
11. </body>
12. </html>
```

home.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. <body>
10. <br><br><br>
11. <center>
12. <a href="addpage">Add Form</a><br><br>
13. <a href="searchpage">Search Form</a><br><br>
14. <a href="updatepage">Update Form</a><br><br>
15. <a href="deletepage">Delete Form</a>
16. </center>
17. </body>
18. </html>
```

addform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br><br>
11. <center>
12. <h2 style="color: red">Durga Software Solutions</h2>
13. <h3 style="color: blue">Student Add Form</h3>
14. <form action="add" method="POST">
15.
16. <table>
17. <tr>
18.   <td>Student Id</td>
19.   <td><input type="text" name="sid"/></td>
20. </tr>
21. <tr>
22.   <td>Student Name</td>
23.   <td><input type="text" name="sname"/></td>
24. </tr>
25. <tr>
26.   <td>Student Address</td>
27.   <td><input type="text" name="saddr"/></td>
28. </tr>
29. <tr>
30.   <td><input type="submit" value="ADD"/></td>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
31. </tr>
32. </table>
33.
34. </form>
35. <center>
36. </body>
37. </html>
```

searchform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <center>
11. <br><br><br>
12. <h2 style="color: red">Durga Software Solutions</h2>
13. <h3 style="color: blue">Student Search Form</h3>
14. <form action="search" method="POST">
15. <table>
16. <tr>
17.   <td>Student Id</td>
18.   <td><input type="text" name="sid"/></td>
19. </tr>
20. <tr>
21.   <td><input type="submit" value="SEARCH"/></td>
22. </tr>
23. </table>
24. </form>
25. </center>
26. </body>
27. </html>
```

updateform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br><br>
11. <center>
12. <h2 style="color: red">Durga Software Solutions</h2>
13. <h3 style="color: blue">Student Update Form</h3>
14. <form action="editform" method="POST">
15. <table>
16. <tr>
17.   <td>Student Id</td>
18.   <td><input type="text" name="sid"/></td>
19. </tr>
20. <tr>
21.   <td><input type="submit" value="GetEditForm"/></td>
22. </tr>
23. </table>
24. </form>
25. </center>
26. </body>
27. </html>
```

student_edit_form.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <br><br><br>
12. <center>
13. <h2 style="color: red">Durga Software Solutions</h2>
14. <h3 style="color: blue">Student Edit Form</h3>
15. <form method="POST" action="update">
16. <table>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
17.<tr>
18. <td>Student Id</td>
19. <td>${sto.sid} <input type="hidden" name="sid" value='${sto.sid}'/></td>
20.</tr>
21.<tr>
22. <td>Student Name</td>
23. <td><input type='text' name='sname' value='${sto.sname}'/></td>
24.</tr>
25.<tr>
26. <td>Student Address</td>
27. <td><input type='text' name='saddr' value='${sto.saddr}'/></td>
28.</tr>
29.<tr>
30. <td><input type="submit" value="UPDATE"></td>
31.</tr>
32.</table>
33.
34.</form>
35.</center>
36.</body>
37.
38.</html>
```

deleteform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br><br>
11. <center>
12. <h2 style="color: red" align="center">Durga Software Solutions</h2>
13. <h3 style="color: blue" align="center">Student Update Form</h3>
14. <form action="delete" method="POST">
15. <table >
16. <tr>
17. <td>Student Id</td>
18. <td><input type="text" name="sid"/></td>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
19.</tr>
20.<tr>
21.  <td><input type="submit" value="DELETE"/></td>
22.</tr>
23.</table>
24.</form>
25.</center>
26.</body>
27.</html>
```

student.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10.<body>
11.<br><br><br>
12.<center>
13.<h2 style="color: red">Durga Software Solutions</h2>
14.<h3 style="color: blue">Student Details</h3>
15.<table border='1'>
16.<tr>
17.  <td>Student Id</td>
18.  <td>${sto.sid}</td>
19.</tr>
20.<tr>
21.  <td>Student Name</td>
22.  <td>${sto.sname}</td>
23.</tr>
24.<tr>
25.  <td>Student Address</td>
26.  <td>${sto.saddr}</td>
27.</tr>
28.</table>
29.<h3>
30.<a href="home">Home Page</a>
31.</h3>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
32.</center>
33.</body>
34.</html>
```

status.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@page isELIgnored="false" %>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <br><br><br>
12. <h2 style="color: red" align="center">
13. ${message}
14. </h2>
15. <h3 align="center">
16. <a href="home">Home Page</a>
17. </h3>
18. </body>
19. </html>
```

HomeController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.AbstractController;
8.
9. public class HomeController extends AbstractController {
10.
11. @Override
12. protected ModelAndView handleRequestInternal(HttpServletRequest arg0, HttpServletResponse arg1) throws Exception {
13.
14. return new ModelAndView("home");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
15. }
16.}
```

StudentAction.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.mutiaction.MultiActionController;
8.
9. import com.durgasoft.dao.StudentDao;
10. import com.durgasoft.dto.StudentTo;
11.
12. public class StudentAction extends MultiActionController {
13.     StudentDao studentDao;
14.     String status = "";
15.     String message = "";
16.
17.     public void setStudentDao(StudentDao studentDao) {
18.         this.studentDao = studentDao;
19.     }
20.
21.     public ModelAndView add(HttpServletRequest request, HttpServletResponse response)
throws Exception{
22.         StudentTo sto = new StudentTo();
23.         sto.setSid(request.getParameter("sid"));
24.         sto.setSname(request.getParameter("sname"));
25.         sto.setSaddr(request.getParameter("saddr"));
26.         status = studentDao.add(sto);
27.         if(status.equals("success")) {
28.
29.             message = "Student Added Successfully";
30.         }
31.         if(status.equals("failure")) {
32.             message = "Student Insertion Failure";
33.         }
34.         if(status.equals("existed")) {
35.             message = "Student Existed Already";
36.         }
37.     return new ModelAndView("status", "message", message);
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
38.    }
39.
40.   public ModelAndView search(HttpServletRequest request, HttpServletResponse response) throws Exception{
41.       String sid = request.getParameter("sid");
42.       StudentTo sto = studentDao.search(sid);
43.       ModelAndView mav = null;
44.       if(sto == null) {
45.           mav = new ModelAndView("status", "message", "Student Not Existed");
46.       }else {
47.           mav = new ModelAndView("student", "sto", sto);
48.       }
49.       return mav;
50.   }
51.
52.   public ModelAndView editform(HttpServletRequest request, HttpServletResponse response) throws Exception{
53.       String sid = request.getParameter("sid");
54.       StudentTo sto = studentDao.search(sid);
55.       ModelAndView mav = null;
56.       if(sto == null) {
57.           mav = new ModelAndView("status", "message", "Student Not Existed");
58.       }else {
59.           mav = new ModelAndView("student_edit_form", "sto", sto);
60.       }
61.       return mav;
62.   }
63.
64.   public ModelAndView update(HttpServletRequest request, HttpServletResponse response) throws Exception{
65.       StudentTo sto = new StudentTo();
66.       sto.setSid(request.getParameter("sid"));
67.       sto.setSname(request.getParameter("sname"));
68.       sto.setSaddr(request.getParameter("saddr"));
69.
70.       String status = studentDao.update(sto);
71.       ModelAndView mav = null;
72.       if(status.equals("success")) {
73.           mav = new ModelAndView("status", "message", "Student Updation Success");
74.       }else {
75.           mav = new ModelAndView("status", "message", "Student Updation Failure");
76.       }
77.       return mav;
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
78.    }
79.
80.   public ModelAndView delete(HttpServletRequest request, HttpServletResponse response
e) throws Exception{
81.     ModelAndView mav = null;
82.     String sid = request.getParameter("sid");
83.     status = studentDao.delete(sid);
84.     if(status.equals("success")){
85.         mav = new ModelAndView("status", "message", "Student Deleted Successfully");
86.     }
87.     if(status.equals("notexisted")){
88.         mav = new ModelAndView("status", "message", "Student Not Existed");
89.     }
90.     if(status.equals("failure")){
91.         mav = new ModelAndView("status", "message", "Student Deletion Failure");
92.     }
93.     return mav;
94.   }
95. }
```

StudentDao.java

```
1. package com.durgasoft.dao;
2.
3. import com.durgasoft.dto.StudentTo;
4.
5. public interface StudentDao {
6.   public String add(StudentTo sto);
7.   public StudentTo search(String sid);
8.   public String update(StudentTo sto);
9.   public String delete(String sid);
10.}
11.
12. StudentDaoImpl.java
13. -----
14. package com.durgasoft.dao;
15.
16. import java.util.List;
17.
18. import org.springframework.jdbc.core.JdbcTemplate;
19.
20. import com.durgasoft.dto.StudentTo;
21.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
22. public class StudentDaoImpl implements StudentDao {  
23.     private JdbcTemplate jdbcTemplate;  
24.     String status = "";  
25.     public void setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
26.         this.jdbcTemplate = jdbcTemplate;  
27.     }  
28.  
29.     @Override  
30.     public String add(StudentTo sto) {  
31.         String query1 = "select * from student where SID = "+sto.getId()+"";  
32.         List<StudentTo> std_List = jdbcTemplate.query(query1, (resultSet, i) -> {  
33.             StudentTo std_Entity = new StudentTo();  
34.             std_Entity.setId(resultSet.getString("SID"));  
35.             std_Entity.setName(resultSet.getString("SNAME"));  
36.             std_Entity.setAddress(resultSet.getString("SADDR"));  
37.             return std_Entity;  
38.         });  
39.         if(std_List.isEmpty()) {  
40.             String query2 = "insert into student values("+sto.getId()+","+sto.getName()+","+  
        sto.getAddress());  
41.             int rowCount = jdbcTemplate.update(query2);  
42.             if(rowCount == 1) {  
43.                 status = "success";  
44.             }else {  
45.                 status = "failure";  
46.             }  
47.         }else {  
48.             status = "existed";  
49.         }  
50.  
51.         return status;  
52.     }  
53.     @Override  
54.     public StudentTo search(String sid) {  
55.         StudentTo sto = null;  
56.         String query = "select * from student where SID = "+sid+"";  
57.         List<StudentTo> std_List = jdbcTemplate.query(query, (resultSet, i) ->{  
58.             StudentTo std_Entity = new StudentTo();  
59.             std_Entity.setId(resultSet.getString("SID"));  
60.             std_Entity.setName(resultSet.getString("SNAME"));  
61.             std_Entity.setAddress(resultSet.getString("SADDR"));  
62.             return std_Entity;  
63.         });  
64.     }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

64.     if(std_List.isEmpty()) {
65.         sto = null;
66.     }else {
67.         sto = std_List.get(0);
68.     }
69.     return sto;
70. }
71.
72. @Override
73. public String update(StudentTo sto) {
74.     String query = "update student set SNAME = "+sto.getSname()+" , SADDR = "+sto.get
    tSaddr()+" where SID = "+sto.getSID()+"";
75.     int rowCount = jdbcTemplate.update(query);
76.     if(rowCount == 1) {
77.         status = "success";
78.     }else {
79.         status = "failure";
80.     }
81.     return status;
82. }
83.
84. @Override
85. public String delete(String sid) {
86.     String query1 = "select * from student where SID = "+sid+"";
87.     List<StudentTo> std_List = jdbcTemplate.query(query1, (resultSet, i) ->{
88.         StudentTo std_Entity = new StudentTo();
89.         std_Entity.setSID(resultSet.getString("SID"));
90.         std_Entity.setSname(resultSet.getString("SNAME"));
91.         std_Entity.setSaddr(resultSet.getString("SADDR"));
92.         return std_Entity;
93.     });
94.     if(std_List.isEmpty()) {
95.         status = "notexisted";
96.     }else {
97.         String query2 = "delete from student where SID = "+sid+"";
98.         int rowCount = jdbcTemplate.update(query2);
99.         if(rowCount == 1) {
100.             status = "success";
101.         }else {
102.             status = "failure";
103.         }
104.     }
105. }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
106.         return status;
107.     }
108. }
```

StudentTo.java

```
1. package com.durgasoft.dto;
2.
3. public class StudentTo {
4.     private String sid;
5.     private String sname;
6.     private String saddr;
7.
8.     public String getSid() {
9.         return sid;
10.    }
11.    public void setSid(String sid) {
12.        this.sid = sid;
13.    }
14.    public String getSname() {
15.        return sname;
16.    }
17.    public void setSname(String sname) {
18.        this.sname = sname;
19.    }
20.    public String getSaddr() {
21.        return saddr;
22.    }
23.    public void setSaddr(String saddr) {
24.        this.saddr = saddr;
25.    }
26.
27.
28.}
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:p="http://www.springframework.org/schema/p"
5.     xmlns:context="http://www.springframework.org/schema/context"
6.     xsi:schemaLocation=""
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7.      http://www.springframework.org/schema/beans
8.      http://www.springframework.org/schema/beans/spring-beans.xsd
9.      http://www.springframework.org/schema/context
10.     http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.    <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
13.      <property name="driverClassName" value="oracle.jdbc.OracleDriver" />
14.      <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe" />
15.      <property name="username" value="system" />
16.      <property name="password" value="durga" />
17.    </bean>
18.
19.    <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
20.      <property name="dataSource" ref="dataSource"/>
21.    </bean>
22.
23.    <bean id="studentDao" class="com.durgasoft.dao.StudentDaoImpl">
24.      <property name="jdbcTemplate" ref="jdbcTemplate"/>
25.    </bean>
26.
27.
28.
29.
30.    <bean name="/home" class="com.durgasoft.controller.HomeController"/>
31.    <bean name="/addpage" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
32.      <property name="viewName" value="addform"/>
33.    </bean>
34.    <bean name="/searchpage" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
35.      <property name="viewName" value="searchform"/>
36.    </bean>
37.    <bean name="/updatepage" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
38.      <property name="viewName" value="updateform"/>
39.    </bean>
40.    <bean name="/deletepage" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
41.      <property name="viewName" value="deleteform"/>
42.    </bean>
43.
44.    <bean name="/" class="com.durgasoft.controller.StudentAction">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
45.   <property name="studentDao" ref="studentDao"/>
46. </bean>
47.
48. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
49.
50. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
51.   <property name="prefix" value="/WEB-INF/"/>
52.   <property name="suffix" value=".jsp"/>
53. </bean>
54.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
13.   <property name="driverClassName" value="oracle.jdbc.OracleDriver" />
14.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe" />
15.   <property name="username" value="system" />
16.   <property name="password" value="durga" />
17. </bean>
18.
19. <bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
20.   <property name="dataSource" ref="dataSource"/>
21. </bean>
22.
23. <bean id="studentDao" class="com.durgasoft.dao.StudentDaoImpl">
24.   <property name="jdbcTemplate" ref="jdbcTemplate"/>
25. </bean>
26.
27.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
28.  
29.  
30. <bean name="/home" class="com.durgasoft.controller.HomeController"/>  
31. <bean name="/addpage" class="org.springframework.web.servlet.mvc.Parameterizable  
ViewController">  
32.   <property name="viewName" value="addform"/>  
33. </bean>  
34. <bean name="/searchpage" class="org.springframework.web.servlet.mvc.Parameterizab  
leViewController">  
35.   <property name="viewName" value="searchform"/>  
36. </bean>  
37. <bean name="/updatepage" class="org.springframework.web.servlet.mvc.Parameteriza  
bleViewController">  
38.   <property name="viewName" value="updateform"/>  
39. </bean>  
40. <bean name="/deletepage" class="org.springframework.web.servlet.mvc.Parameterizabl  
eViewController">  
41.   <property name="viewName" value="deleteform"/>  
42. </bean>  
43.  
44. <bean name="/*" class="com.durgasoft.controller.StudentAction">  
45.   <property name="studentDao" ref="studentDao"/>  
46. </bean>  
47.  
48. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanN  
ameUrlHandlerMapping"/>  
49.  
50. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalReso  
urceViewResolver">  
51.   <property name="prefix" value="/WEB-INF//>  
52.   <property name="suffix" value=".jsp"/>  
53. </bean>  
54.</beans>
```

To run the above application we need the following jar files in web application lib folder.

- commons-logging-1.2.jar
- ojdbc6.jar
- spring-aop-4.3.9.RELEASE.jar
- spring-aspects-4.3.9.RELEASE.jar
- spring-beans-4.3.9.RELEASE.jar
- spring-context-4.3.9.RELEASE.jar
- spring-context-support-4.3.9.RELEASE.jar
- spring-core-4.3.9.RELEASE.jar

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- spring-expression-4.3.9.RELEASE.jar
- spring-jdbc-4.3.9.RELEASE.jar
- spring-tx-4.3.9.RELEASE.jar
- spring-web-4.3.9.RELEASE.jar
- spring-webmvc-4.3.9.RELEASE.jar

3. Command Controllers:

Command Class:

Command Class is a normal Java Bean class, it can be instantiate by Spring WEB MVC framework inorder to store form data which is submitted by the respective Client along with request.

The main intention to store form data in Command Class objects is

1. To make available form data to Business Logic to use.
2. To transfer form data from Controller Layer to View Layer like DTO[Data Transfer Object]
3. To perform Server side Data Validations before using data in Business logic.

To use Command classes in Spring WEB MVC applications, we have to use the following conventions.

1. Command class must be a normal JAVA Bean class.
2. Command class must be a public, non-abstract and non-final class.
3. Command class must have the properties whose names must be same as the form properties name.
4. In Command class, we must provide a separate setXXX() method and getXXX() method for each and every property.
5. In Command classes we have to declare all properties as private and methods as public.
6. If we want to provide constructor in Command class then we can provide constructor, but, it must be public and 0-argument constructor.
7. It is suggested to implement java.io.Serializable interface

EX:

```
1. public class Student implements Serializable{  
2.     private String sid;  
3.     private String sname;  
4.     private String saddr;  
5.     setXXX() and getXXX()  
6. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: In Spring WEB MVC applications, Framework will create a separate Command object for each and every request which is generated from form.

If we want to use Command classes in Spring WEB MVC Applications then we have to use Command Controller Classes.

Spring has provided the following Command Controller classes to use Command classes.

1. BaseCommandController
2. AbstractCommandController
3. AbstractFormController
4. SimpleFormController
5. AbstractWizardFormController

1. **BaseCommandController:**

It is an abstract class provided by SpringFramework as "**org.springframework.web.servlet.mvc.BaseCommandController**".

It is a base class for all Command Controller classes which are wishing to populate request parameters data in Command class objects.

Note: It is a deprecated abstract class , it was not existed in Spring4.x version.

2. **AbstractCommandController:**

It is an abstract class provided by Spring Framework as "**org.springframework.web.servlet.mvc.AbstractCommandController**" with the following methods.

- protected abstract ModelAndView handle(HttpServletRequest request, HttpServletResponse response, java.lang.Object command, BindException errors)
- protected ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response)

This controller class will populate form data in Command class objects automatically by creating Command class object for each and every request.

In general, we will use Controller class when we have form submission from client and when we dont want to perform Data Validations at Server side.

If we want to use this CommandController class then we have to declare an user defined class and it must be extended from "AbstractCommandController" abstract class and we must override handle(--) method.

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: In User defined Controller class , we have to provide 0-arg constructor, where we have to set command class by using the following method.

```
public void setCommandClass(Class class)
```

or

set the following properties in Controller class configuration in Spring configuration file.

1.commandName: any name

2.commandClass: Fully qualified name of the Command class.

EX:

```
<bean name="/login" class="com.durgasoft.controller.LoginController">
    <property name="commandName" value="user"/>
    <property name="commandClass" value="com.durgasoft.command.User"/>
</bean>
```

Note: AbstractCommandController class is deprecated and it was removed from Spring4.x version, so that, to execute the below application we have to use either Spring2.5 versin or atleast Spring3.x version.

EX:

index.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="loginpage"/>
11. </body>
12. </html>
```

loginform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red">Durga Software Solutions</h2>
11. <h3 style="color: blue">User Login Page</h3>
12. <form method="POST" action="login">
13. <table>
14. <tr>
15.   <td>User Name</td>
16.   <td><input type="text" name="uname"/></td>
17. </tr>
18. <tr>
19.   <td>Password</td>
20.   <td><input type="password" name="upwd"/></td>
21. </tr>
22. <tr>
23.   <td><input type="submit" value="Login"/></td>
24. </tr>
25. </table>
26. </form>
27. </body>
28. </html>
```

status.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red">Durga Software Solutions</h2>
11. <h3 style="color: green">User Logic Status </h3>
12. <h2 style="color: blue">${message}</h2>
13. <h3>
14. <a href="loginpage">Login Form</a>
15. </h3>
16. </body>
17. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

LoginController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.validation.BindException;
7. import org.springframework.web.servlet.ModelAndView;
8. import org.springframework.web.servlet.mvc.AbstractCommandController;
9.
10.
11. import com.durgasoft.command.User;
12.
13. public class LoginController extends AbstractCommandController{
14.     @Override
15.     protected ModelAndView handle(HttpServletRequest request, HttpServletResponse response, Object command, BindException exception)
16.             throws Exception {
17.         User user = (User)command;
18.         String uname = user.getUsername();
19.         String upwd = user.getPassword();
20.         ModelAndView mav = null;
21.         if(uname.equals("durga") && upwd.equals("durga")) {
22.             mav = new ModelAndView("status", "message", "User Login Success");
23.         }else {
24.             mav = new ModelAndView("status", "message", "User Login Failure");
25.         }
26.         return mav;
27.     }
28. }
```

User.java

```
1. package com.durgasoft.command;
2.
3. import java.io.Serializable;
4.
5. public class User implements Serializable{
6.     private String uname;
7.     private String upwd;
8.
9.     public String getUsername() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10.    return uname;
11. }
12. public void setUname(String uname) {
13.     this.uname = uname;
14. }
15. public String getUpwd() {
16.     return upwd;
17. }
18. public void setUpwd(String upwd) {
19.     this.upwd = upwd;
20. }
21.}
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.
13. <bean name="/loginpage" class="org.springframework.web.servlet.mvc.Parameterizable
  ViewController">
14.   <property name="viewName" value="loginform"/>
15. </bean>
16. <bean name="/login" class="com.durgasoft.controller.LoginController">
17.   <property name="commandName" value="user"/>
18.   <property name="commandClass" value="com.durgasoft.command.User"/>
19. </bean>
20.
21. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanN
  ameUrlHandlerMapping"/>
22.
23.<bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceVi
  ewResolver">
24. <property name="prefix" value="/WEB-INF/"/>
25. <property name="suffix" value=".jsp"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
26.</bean>
27.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app7</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```

3. AbstractFormController

It is an abstract class provided by Spring Framework as

"**org.springframework.web.servlet.mvc.AbstractFormController**" with the following methods.

- ❖ protected ModelAndView processFormSubmission(HttpServletRequest request, HttpServletResponse response, Object command, BindException exception) throws Exception {
- ❖ protected ModelAndView showForm(HttpServletRequest request, HttpServletResponse response, BindException exception) throws Exception {
- ❖ Where processFormSubmission() method will handle the request , It will include the application logic which we want to execute after submmitting form, It will be executed when we submit POST rquest from the user form.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ❖ Where we have to override showForm() method to prepare view name and it will be executed when we submitted GET request from client.

This controller class will populate form data in Command class objects automatically either by creating Command class object for each and every request or it will reuse the command class object from session scope if we set "sessionForm" property value true.

In general, we will use Controller class when we have form submission from client and when we don't want to perform Data Validations at Server side.

If we want to use this CommandController class then we have to declare an user defined class and it must be extended from "AbstractCommandController" abstract class and we must override processFormSubmission(--) method and showForm() method.

Note: In User defined Controller class , we have to provide 0-arg constructor, where we have to set command class by using the following method.

```
public void setCommandClass(Class class)  
or
```

We have to set the following properties in Controller bean configurations in spring configuration file.

1. sessionForm --> true
2. commandName --> any name
3. commandClass --> Fully qualified name of the command class.

Note: AbstractFormController class is deprecated in Spring3.x version, to use this Controller class we have to use either Spring2.5 version atleast Spring3.x version.

Example:

index.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
2. pageEncoding="ISO-8859-1"%>  
3. <!DOCTYPE html>  
4. <html>  
5. <head>  
6. <meta charset="ISO-8859-1">  
7. <title>Insert title here</title>  
8. </head>  
9. <body>  
10. <jsp:forward page="reg"/>  
11. </body>  
12. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

registrationform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red" align="center">Durga Software Solutions</h2>
11. <h3 style="color: blue" align="center">User Registration Page </h3>
12. <form method="POST" action="reg">
13. <center>
14. <table>
15. <tr>
16.   <td>Student Id</td>
17.   <td><input type="text" name="sid"/></td>
18. </tr>
19. <tr>
20.   <td>Student Name</td>
21.   <td><input type="text" name="sname"/></td>
22. </tr>
23. <tr>
24.   <td>Student Email</td>
25.   <td><input type="text" name="semail"/></td>
26. </tr>
27. <tr>
28.   <td>Student Mobile</td>
29.   <td><input type="text" name="smobile"/></td>
30. </tr>
31. <tr>
32.   <td><input type="submit" value="Registration"/></td>
33. </tr>
34. </table>
35. </center>
36. </form>
37. </body>
38. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

registrationdetails.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
11. <h3 style="color: blue;" align="center">Student Registration Details</h3>
12. <center>
13. <table border='1'>
14. <tr>
15.   <td>Student Id</td>
16.   <td>${student.sid}</td>
17. </tr>
18. <tr>
19.   <td>Student Name</td>
20.   <td>${student.sname}</td>
21. </tr>
22. <tr>
23.   <td>Student Email Id</td>
24.   <td>${student.semail}</td>
25. </tr>
26. <tr>
27.   <td>Student Mobile No</td>
28.   <td>${student.smobile}</td>
29. </tr>
30. </table>
31. </center>
32. </body>
33. </html>
```

Student.java

```
1. package com.durgasoft.command;
2.
3. public class Student {
4.   private String sid;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. private String sname;
6. private String semail;
7. private String smobile;
8.
9. public String getSid() {
10.     return sid;
11. }
12. public void setSid(String sid) {
13.     this.sid = sid;
14. }
15. public String getSname() {
16.     return sname;
17. }
18. public void setSname(String sname) {
19.     this.sname = sname;
20. }
21. public String getSEmail() {
22.     return semail;
23. }
24. public void setSEmail(String semail) {
25.     this.semail = semail;
26. }
27. public String getSmobile() {
28.     return smobile;
29. }
30. public void setSmobile(String smobile) {
31.     this.smobile = smobile;
32. }
33. }
```

StudentController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.validation.BindException;
7. import org.springframework.web.servlet.ModelAndView;
8. import org.springframework.web.servlet.mvc.AbstractFormController;
9.
10.
11. import com.durgasoft.command.Student;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12.
13. public class StudentController extends AbstractFormController {
14.
15.     @Override
16.     protected ModelAndView processFormSubmission(HttpServletRequest request, HttpServletRe
    sponse response, Object command,
17.             BindException exception) throws Exception {
18.         Student student = (Student)command;
19.         ModelAndView mav = new ModelAndView("registrationdetails", "student", student);
20.         return mav;
21.     }
22.
23.     @Override
24.     protected ModelAndView showForm(HttpServletRequest arg0, HttpServletResponse ar
    g1, BindException arg2)
25.         throws Exception {
26.
27.         return new ModelAndView("registrationform");
28.     }
29.

```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.         xmlns:p="http://www.springframework.org/schema/p"
5.         xmlns:context="http://www.springframework.org/schema/context"
6.         xsi:schemaLocation="
7.             http://www.springframework.org/schema/beans
8.             http://www.springframework.org/schema/beans/spring-beans.xsd
9.             http://www.springframework.org/schema/context
10.            http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.     <bean name="/reg" class="com.durgasoft.controller.StudentController">
13.         <property name="sessionForm" value="true"/>
14.         <property name="commandName" value="student"/>
15.         <property name="commandClass" value="com.durgasoft.command.Student"/>
16.     </bean>
17.     <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
18.     <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

19. <property name="prefix" value="/WEB-INF/" />
20. <property name="suffix" value=".jsp" />
21. </bean>
22. </beans>
```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3. <display-name>app11</display-name>
4. <welcome-file-list>
5. <welcome-file>index.html</welcome-file>
6. <welcome-file>index.htm</welcome-file>
7. <welcome-file>index.jsp</welcome-file>
8. <welcome-file>default.html</welcome-file>
9. <welcome-file>default.htm</welcome-file>
10. <welcome-file>default.jsp</welcome-file>
11. </welcome-file-list>
12. <servlet>
13. <servlet-name>ds</servlet-name>
14. <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15. <load-on-startup>1</load-on-startup>
16. </servlet>
17. <servlet-mapping>
18. <servlet-name>ds</servlet-name>
19. <url-pattern>/</url-pattern>
20. </servlet-mapping>
21. </web-app>
```

**Spring MVC Tag Library:**

In Spring WEB MVC based applications, to prepare Presentation or View part we are able to use basic View technologies like HTML, JSP, JSTL , Velocity , Free Marker,.....

In Spring WEB MVC based applications, to prepare User Forms Spring WEB MVC framework has given a separate tag library.

To use Spring WEB MVC web applications if we want to use Spring provided tag library then we have to use keep spring-webmvc.jar in web application lib folder and we have to use the following URL in "uri" attribute in <taglib> directive.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

<http://www.springframework.org/tags/form>**EX:** <%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>

Spring has provided the following set of form tags to prepare User forms.

1.<form:form>

It is same as html <form> tag, it will gather/group all other Html components to take data from users inorder to submit data to Server side application.

Syntax:

```
<form:form method="—" action="—" commandName="---">  
</form:form>
```

- ❖ Where "method" attribute will take HTTP-Method like GET or POST,....
- ❖ Where "action" attribute will take action URI of the respective controller class.
- ❖ Where "commandName" attribute will take Command class object logical name , which must be same as the value which is specified along with "commandName" property in controller class configuration in Spring configuration file inorder to store form data in the Command class object.

EX:**registrationform.jsp**

```
<form:form method="POST" action="reg" commandName="user">  
</form:form>
```

User.java

```
public class User{  
----  
}
```

ds-servlet.xml

```
<beans>  
----  
<bean name="reg" class="com.durgasoft.controller.RegistrationController">  
  <property name="commandName" value="user"/>  
  <property name="commandClass" value="com.durgasoft.command.User"/>  
----
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
</bean>
-----
</beans>
```

2. <form:input>

This tag is same as HTML <input type="text"/> tag, it will create textfield in user form , it is an input GUI component, it will take a line text data from Users.

Syntax:

```
<form:input path="--" size="--" value="--"/>
```

- ❖ Where "path" attribute will take reference name to the text field and it must be same as a property in Command class having setter and getter method.
- ❖ Where "size" attribute will take textfield size.
- ❖ Where "value" attribute will take a default message to display while preparing text field.

```
<form:form .... >
User Name<form:input path="uname" size="20"/>
</form:form>
```

In Command class, we must declare "uname" as String property.

```
public class User{
```

```
----
private String uname;
----
public void setUserName(String uname){
this.uname = uname;
}
public String getUserName(){
return uname;
}
----
```

3. <form:password>

It will generate password field in user form, it is same as "<input type="password"/>" tag, it will take password data from users.

Syntax:

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<form:password path="--" value="--" showPassword="--"/>
```

- ❖ Where "path" attribute will take reference name to the password field, it must be same as the property defined in the respective Command class having setter and getter methods.
- ❖ Where "value" attribute will take default password value to display while creating password field.
- ❖ Where "showPassword" property will take either true/false value to show or not to show password data in password field while providing password data.

EX:

```
<form:form ----->
User Password<form:password name="upwd" showPassword="false"/>
</form:form>
```

In Command class:

```
public class User{
private String upwd;
---
public void setUpwd(String upwd){
this.upwd = upwd;
}
public String getUpwd(){
return upwd;
}
---}
```

Note: "showPassword" attribute is not working in Applications as part of my testing, but, Spring Framework has provided "showPassword" attribute in documentation.

4. **<form:checkbox>**

It will provide checkbox in user form, it is same as Html `<input type="checkbox">` tag and it will submit a boolean value to the Server side application on the basis of the selection. If we select checkbox then it will send "true" value , if we are not selecting checkbox then it will send false value to server side application.

Syntax:

```
<form:checkbox path="--" value="--"/>
```

- ❖ Where "path" attribute will take reference name to the checkbox, it must be same as the boolean property existed in Command class having setXXX() and isXXX() method.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786
Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ❖ Where "value" will take a value to send to the Server side application, where at Server side Command class we need a property of String or xxx data type with setXXX() and getXXX() method.

Ex:

```
<form:form ---->
  R U Married? <form:checkbox path="Married"/>
</form:form>
```

In Command Class:

```
public class User{
    private boolean married;
    ----
    public void setMarried(boolean married){
        this.married = married;
    }
    public boolean isMarried(){
        return married;
    }
}
```

EX:

```
<form:form --- >
  R u Married? <form:checkbox path="maritalStatus" value="Married"/>
</form:form>
```

In Command Class:

```
public class User{
    private String maritalStatus;
    ----
    public void setMaritalStatus(String maritalStatus) {
        this.maritalStatus = maritalStatus;
    }
    public String getMaritalStatus() {
        if(maritalStatus == null || maritalStatus.equals("")) {
            maritalStatus="Not Married";
        }
        return maritalStatus;
    }
    ----
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

5. <form:checkboxes>

It will provide collection of checkboxes to select items.

Syntax:

```
<form:checkboxes path="--" items="--"/>
```

- ❖ Where "path" attribute will take reference value of all checkboxes , it must have a property of String[] in the respective command class having setXXX() and getXXX() methods.
- ❖ Where "items" attribute will take an expression including a property from Command class providing checkbox labels and values or a Property data which is generated by overriding referenceData() method in controller class.

EX:

```
<form:form ---- >  
User Qualifications<form:checkboxes path="uqual" items="{qual_List}" />  
</form:form>
```

In Controller class:

```
public class RegistrationController extends SimpleFormController{  
  
    @Override  
    protected Map referenceData(HttpServletRequest request) throws Exception {  
        Map<String, Object> map = new HashMap<>();  
        List<String> qual_List = new ArrayList<>();  
        qual_List.add("BSC");  
        qual_List.add("BTech");  
        qual_List.add("MCA");  
        qual_List.add("MTech");  
        qual_List.add("PHD");  
        map.put("qual_List", qual_List);  
        return map;  
    }  
  
    @Override  
    protected ModelAndView onSubmit(Object command) throws Exception {  
        User user = (User)command;  
        return new ModelAndView("status", "user", user);  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In Command Class:

```
public class User{  
    private String[] uqual;  
    ---  
    public void setUqual(String[] uqual){  
        this.uqual = uqual;  
    }  
    public String getUqual(){  
        return uqual;  
    }  
}
```

6. <form:radiobutton>

It will display a radio button to select an item, it is same as html
<input type="radio"> tag.

Syntax:

```
<form:radiobutton path="--" value="--"/>
```

- ❖ Where "path" attribute will take a reference name , it must be same as the property in Command class having setXXX() method and getXXX() method.
- ❖ Where "value" attribute will take radio button value inorder to send to Server side application.

EX:

```
<form:radiobutton path="ugender" value="Male"/>Male  
<form:radiobutton path="ugender" value="Female"/>Female
```

In Command class:

```
public class User{  
    private String ugender;  
    ---  
    public void setUgender(String ugender){  
        this.ugender = ugender;  
    }  
    public String getUgender(){  
        return ugender;  
    }  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

7. <formradioButtons>

It will provide multiple radio buttons as a group to select one item.

Syntax:

```
<form:radioButtons path="--" items="--"/>
```

- ❖ Where "path" attribute will take reference name to the radio buttons which same as the property name in command class inorder to store the selected radio button value.
- ❖ Where "items" attribute will take the property name of the List or String[] which contains all the names of the Radio Buttons which is generated from referencedData() method from the respective controller class.

EX:

```
<form:radioButtons path="uworkLocation" items="${uworkLocation}"/>
```

In Command class:

```
public class User{  
private String uworkLocation;  
----  
public void setUworkLocation(String uworkLocation){  
this.uworkLocation = uworkLocation;  
}  
public String getUworkLocation(){  
return uworkLocation;  
}  
}
```

In Controller class:

```
public class UserController extends SimpleFormController{  
protected Map referenceData(HttpServletRequest request) throws Exception {  
Map<String, Object> map = new HashMap<>();  
  
List<String> uworkLocation = new ArrayList<>();  
uworkLocation.add("Hyderabad");  
uworkLocation.add("Chennai");  
uworkLocation.add("Pune");  
uworkLocation.add("Mumbai");  
map.put("uworkLocation", uworkLocation);  
  
return map;  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

}

8. <form:select>

<form:select> tag can be used to create select box, it is same as html <select> tag, it will display list of items to select either multiple or single depending on our option.

Syntax:

```
<form:select path="--" multiple="--" items="--">  
-----  
</form:select>
```

- ❖ Where "path" attribute will take reference value to the Select box , it must be same as the respective property name in command class inorder to store the selected values.
- ❖ Where "multiple" attribute will take either "true" or "false" value inorder make "multiple" selections or "single" selection in the select box.
- ❖ Where "items" attribute will take a property name of Map type contains the Item values and itemLabels which is generated from referencedData() method of the controller class.

Note: If we want to provide items to the select box individually , not as a Map from controller class then we have to use <form:option> tag as child tag to <form:select> tag.

Syntax:

```
<form:option value="--"> Label </form:option>
```

Where "value" attribute will take item Value which we want to send to Server side application.
Where "Label" is item Label to display in select box.

Note: If we want to get all the Item Values and Item Labels from referencesData() method in Controller class then we have to use <form:options> tag.

Syntax:

```
<form:options items="--" itemValue="--" itemLabel="--"/>
```

Where "items" attribute will take a property representing Map object contains item values and item Labels from referencedData() method in Controller class.

EX:

```
<form:select path="uskillSet" multiple="true">  
    <form:option value="C">C</form:option>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
<form:option value="C++">C++</form:option>
<form:option value="Java">Java</form:option>
<form:option value=".Net">.Net</form:option>
<form:option value="Oracle">Oracle</form:option>
</form:select>
```

In Command Class:

```
public class User{
private String[] uskillSet;
-----
public void setUskillSet(String[] uskillSet){
this.uskillSet = uskillSet;
}
public String[] getUskillSet(){
return uskillSet;
}
}
```

EX:

```
<form:select path="uhobbies" items="${uhobbies}" multiple="true"/>
```

In Command Class:

```
public class User{
private String[] uhobbies;
-----
public void setUhobbies(String[] uhobbies){
this.uhobbies = uhobbies;
}
public String[] getUhobbies(){
return uhobbies;
}
}
```

In Controller class:

```
public class UserController extends SimpleFormController{
protected Map referenceData(HttpServletRequest request) throws Exception {
Map<String, Object> map = new HashMap<>();
Map<String, String> uhobbies = new HashMap<>();
uhobbies.put("Playing Cricket", "Playing Cricket");
uhobbies.put("Listening Music", "Listening Music");
}
```

CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28**US NUM:** 4433326786**Mail ID:** durgasoftonlinetraining@gmail.com**WEBSITE:** www.durgasoftonline.com**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
uhobbies.put("Chatting with Friends", "Chatting with Friends");
map.put("uhobbies", uhobbies);

map.put("uhobbies", uhobbies);

return map;
}
```

EX:

```
<form:select path="uprofession" multiple="false">
    <form:options items="${uprofession}" />
</form:select>
```

IN Command Class:

```
public class User{
private String uprofession;
---

public void setUprofession(String uprofession){
this.profession = profession;
}
public String getUpofession(){
return uprofession;
}
}
```

9. <form:textarea>

It will provide text area to take multiple lines of data from Users, it is same as <textarea> tag in Html.

Syntax:

```
<form:textarea path="--"/>
```

Where "path" attribute will take reference name to the textarea which is same as a property in Command class.

EX:

```
<form:textarea path="uaddr"/>
```

In Command class:

```
public class User{
private String uaddr;
---
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public void setUaddr(String uaddr){  
    this.uaddr = uaddr;  
}  
public String getUaddr(){  
    return uaddr;  
}
```

10. <form:hidden>

It will prepare Hidden field in user forms, it is same as <input type="hidden"/> in Html.

Syntax:

```
<form:hidden path="--" value="--"/>
```

Where "path" attribute will take a reference name to the hidden field, it must be same as a property in Command class having setXXX() method and getXXX() method.

EX:

```
<form:hidden path="sid" value="S-111"/>
```

In Command Class:

```
public class Student{  
private String sid;  
public void setSid(String sid){  
this.sid = sid;  
}  
public String getSid(){  
return sid;  
}
```

4. SimpleFormController:

This controller class is able to support the Command classes inorder to store form data in Command class objects when we submit user forms.

It was introduced in Spring2.5 version as
"org.springframework.web.servlet.mvc.SimpleFormController" and it was deprecated in SPring3.x
version and it was removed in spring4.x version.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

It is best option for Data Validations in Spring WEB MVC Framework, it will render the same page automatically when validation errors are identified and it will forward to successView page when no Validation messages are identified.

If we want to use this controller class then we have to declare an user defined class and it must be extended from SimpleFormController class and we must implement either of the following methods.

protected ModelAndView onSubmit(Object command) throws Exception

protected ModelAndView onSubmit(HttpServletRequest request, HttpServletResponse response, Object command, BindingException exception) throws Exception

onSubmit(--) method will take the application logic which we want to execute by submitting form from client.

If we want to use this Controller class, we must provide the the following properties in Controller class configuration.

1. **formView:** it will take form name to get User form.
2. **commandName:** it will take logical name to the command class.
3. **commandClass:** It will take fully qualified name of the command class.
4. **sessionForm:** to keep Command Object in session scope inorder to reuse command object.

In case of SimpleFormController, if we want to provide data to the GUI components like checkboxes, radio buttons, Select boxes,..... in user forms from Controller class then we have to override the following method from SimpleFormController class.

public Map referencedData(HttpServletRequest request) throws Exception

Example:

index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10.<jsp:forward page="reg"></jsp:forward>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11.</body>
12.</html>
```

registrationform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <h2 style="color: red" align="center">Durga Software Solutions</h2>
12. <h3 style="color: blue" align="center">User Registration Page</h3>
13. <form:form method="POST" action="reg" commandName="user">
14. <center>
15. <table style="background: lightyellow;">
16. <tr>
17.   <td>User Name</td>
18.   <td><form:input path="uname" size="20" value="User Name"/></td>
19. </tr>
20. <tr>
21.   <td>Password</td>
22.   <td><form:password path="upwd" value="abc123" showPassword="true"/></td>
23. </tr>
24. <tr>
25.   <td>R U Married?</td>
26.   <td><form:checkbox path="maritalStatus1"/></td>
27. </tr>
28. <tr>
29.   <td>R U Single?</td>
30.   <td><form:checkbox path="maritalStatus2" value="Single"/></td>
31. </tr>
32. <tr>
33.   <td>User Qualifications</td>
34.   <td><form:checkboxes items="${qual_List}" path="uqual" /></td>
35. </tr>
36. <tr>
37.   <td>User Gender</td>
38.   <td>
```

CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28**US NUM:** 4433326786**Mail ID:** durgasoftonlinetraining@gmail.com**WEBSITE:** www.durgasoftonline.com**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
39.   <form:radioButton path="ugender" value="Male"/>Male
40.   <form:radioButton path="ugender" value="Female"/>Female
41. </td>
42.</tr>
43.<tr>
44.   <td>User Work Location</td>
45.   <td>
46.     <form:radiobuttons path="uworkLocation" items="${uworkLocation}"/>
47.   </td>
48.</tr>
49.<tr>
50.   <td>User Skill Set</td>
51.   <td>
52.     <form:select path="uskillSet" multiple="true">
53.       <form:option value="C">C</form:option>
54.       <form:option value="C++">C++</form:option>
55.       <form:option value="Java">Java</form:option>
56.       <form:option value=".Net">.Net</form:option>
57.       <form:option value="Oracle">Oracle</form:option>
58.     </form:select>
59.   </td>
60.</tr>
61.<tr>
62.   <td>User Hobbies</td>
63.   <td><form:select path="uhobbies" items="${uhobbies}" multiple="true"/></td>
64.</tr>
65.<tr>
66.   <td>User Profession</td>
67.   <td>
68.     <form:select path="uprofession" multiple="false">
69.       <form:options items="${uprofession}"/>
70.     </form:select>
71.   </td>
72.</tr>
73.<tr>
74.   <td>User Address</td>
75.   <td>
76.     <form:textarea path="uaddr"/>
77.   </td>
78.</tr>
79.<tr>
80.   <td><input type="submit" value="Registration"/></td>
81.</tr>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

82.</table>
83.</center>
84.<form:form>
85.</body>
86.</html>
```

status.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
4. <!DOCTYPE html>
5. <html>
6. <head>
7. <meta charset="ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10.<body>
11.<h2 style="color: red" align="center">Durga Software Solutions</h2>
12.<h3 style="color: green" align="center">User Registration Status </h3>
13.<center>
14.<table border="1" style="background: lightyellow;">
15.<tr>
16.  <td>User Name</td>
17.  <td>${user.uname}</td>
18.</tr>
19.<tr>
20.  <td>User Password</td>
21.  <td>${user.upwd}</td>
22.</tr>
23.<tr>
24.  <td>R U Married?</td>
25.  <td>${user.maritalStatus1}</td>
26.</tr>
27.<tr>
28.  <td>R U Single?</td>
29.  <td>${user.maritalStatus2}</td>
30.</tr>
31.<tr>
32.  <td>User Qualifications</td>
33.  <td>
34.    <c:forEach var="qual" items="${user.uqual}">
35.      <c:out value="${qual}"/><br>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
36.    </c:forEach>
37.  </td>
38.<tr>
39.<tr>
40.  <td>User Gender</td>
41.  <td>${user.ugender}</td>
42.</tr>
43.<tr>
44.  <td>User Work Location</td>
45.  <td>${user.uworkLocation}</td>
46.</tr>
47.<tr>
48.  <td>User Skill Set</td>
49.  <td>
50.    <c:forEach var="skill" items="${user.uskillSet}">
51.      <c:out value="${skill}" /><br>
52.    </c:forEach>
53.  </td>
54.</tr>
55.<tr>
56.  <td>User Hobbies</td>
57.  <td>
58.    <c:forEach var="hobbit" items="${user.uhobbies}">
59.      <c:out value="${hobbit}" /><br>
60.    </c:forEach>
61.  </td>
62.</tr>
63.<tr>
64.  <td>User Profession</td>
65.  <td>${user.uprofession}</td>
66.</tr>
67.<tr>
68.  <td>User Address</td>
69.  <td>${user.uaddr}</td>
70.</tr>
71.</table>
72.</center>
73.<h3 align="center">
74. <a href="reg">User Registration Form</a>
75.</h3>
76.
77.</body>
78.</html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

User.java

```
1. package com.durgasoft.command;
2.
3. import java.io.Serializable;
4.
5. public class User implements Serializable{
6.     private String uname;
7.     private String upwd;
8.     private boolean maritalStatus1;
9.     private String maritalStatus2;
10.    private String[] uqual;
11.    private String ugender;
12.    private String uworkLocation;
13.    private String uskillSet;
14.    private String[] uhobbies;
15.    private String uprofession;
16.    private String uaddr;
17.
18.
19.    public String getUname() {
20.        return uname;
21.    }
22.    public void setUname(String uname) {
23.        this.uname = uname;
24.    }
25.    public String getUpwd() {
26.        return upwd;
27.    }
28.    public void setUpwd(String upwd) {
29.        this.upwd = upwd;
30.    }
31.
32.    public void setMaritalStatus1(boolean maritalStatus1) {
33.        this.maritalStatus1 = maritalStatus1;
34.    }
35.    public boolean isMaritalStatus1() {
36.        return maritalStatus1;
37.    }
38.
39.    public void setMaritalStatus2(String maritalStatus2) {
40.        this.maritalStatus2 = maritalStatus2;
41.    }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
42. public String getMaritalStatus2() {  
43.     if(maritalStatus2 == null || maritalStatus2.equals("")) {  
44.         maritalStatus2 = "Married";  
45.     }  
46.     return maritalStatus2;  
47. }  
48.  
49. public void setUqual(String[] uqual) {  
50.     this.uqual = uqual;  
51. }  
52. public String[] getUqual() {  
53.     return uqual;  
54. }  
55.  
56. public void setUgender(String ugender) {  
57.     this.ugender = ugender;  
58. }  
59. public String getUgender() {  
60.     return ugender;  
61. }  
62.  
63. public void setUworkLocation(String uworkLocation) {  
64.     this.uworkLocation = uworkLocation;  
65. }  
66. public String getUworkLocation() {  
67.     return uworkLocation;  
68. }  
69.  
70. public void setUskillSet(String uskillSet) {  
71.     this.uskillSet = uskillSet;  
72. }  
73. public String getUskillSet() {  
74.     return uskillSet;  
75. }  
76.  
77. public void setUhobbies(String[] uhobbies) {  
78.     this.uhobbies = uhobbies;  
79. }  
80. public String[] getUhobbies() {  
81.     return uhobbies;  
82. }  
83.  
84. public void setUprofession(String uprofession) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
85.     this.uprofession = uprofession;
86. }
87. public String getUprofession() {
88.     return uprofession;
89. }
90.
91. public void setUaddr(String uaddr) {
92.     this.uaddr = uaddr;
93. }
94. public String getUaddr() {
95.     return uaddr;
96. }
97.}
```

RegistrationController.java



```
1. package com.durgasoft.controller;
2.
3. import java.util.ArrayList;
4. import java.util.HashMap;
5. import java.util.List;
6. import java.util.Map;
7.
8. import javax.servlet.http.HttpServletRequest;
9. import javax.servlet.http.HttpServletResponse;
10.
11. import org.springframework.validation.BindException;
12. import org.springframework.web.servlet.ModelAndView;
13. import org.springframework.web.servlet.mvc.SimpleFormController;
14.
15. import com.durgasoft.command.User;
16.
17. public class RegistrationController extends SimpleFormController{
18.
19.     @Override
20.     protected Map referenceData(HttpServletRequest request) throws Exception {
21.         Map<String, Object> map = new HashMap<>();
22.
23.         List<String> qual_List = new ArrayList<>();
24.         qual_List.add("BSC");
25.         qual_List.add("BTech");
26.         qual_List.add("MCA");
27.         qual_List.add("MTech");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

28.     qual_List.add("PHD");
29.     map.put("qual_List", qual_List);
30.
31.     List<String> uworkLocation = new ArrayList<>();
32.     uworkLocation.add("Hyderabad");
33.     uworkLocation.add("Chennai");
34.     uworkLocation.add("Pune");
35.     uworkLocation.add("Mumbai");
36.     map.put("uworkLocation", uworkLocation);
37.
38.     Map<String, String> uhobbies = new HashMap<>();
39.     uhobbies.put("Playing Cricket", "Playing Cricket");
40.     uhobbies.put("Listening Music", "Listening Music");
41.     uhobbies.put("Chatting with Friends", "Chatting with Friends");
42.     map.put("uhobbies", uhobbies);
43.
44.     Map<String, String> uprofession = new HashMap<>();
45.     uprofession.put("Former", "Former");
46.     uprofession.put("Employee", "Employee");
47.     uprofession.put("Teacher", "Teacher");
48.     map.put("uprofession", uprofession);
49.
50.     return map;
51. }
52.
53. @Override
54. protected ModelAndView onSubmit(Object command) throws Exception {
55.     User user = (User)command;
56.     return new ModelAndView("status", "user", user);
57. }
58.

```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.         xmlns:p="http://www.springframework.org/schema/p"
5.         xmlns:context="http://www.springframework.org/schema/context"
6.         xsi:schemaLocation="
7.             http://www.springframework.org/schema/beans
8.             http://www.springframework.org/schema/beans/spring-beans.xsd
9.             http://www.springframework.org/schema/context"

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.    <!--
13.    <bean name="/registrationpage" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
14.        <property name="viewName" value="registrationform"/>
15.    </bean>
16.    -->
17.    <bean name="/reg" class="com.durgasoft.controller.RegistrationController">
18.        <property name="formView" value="registrationform"/>
19.        <property name="sessionForm" value="true"/>
20.        <property name="commandName" value="user"/>
21.        <property name="commandClass" value="com.durgasoft.command.User"/>
22.    </bean>
23.
24.    <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
25.
26.    <bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
27.        <property name="prefix" value="/WEB-INF/"/>
28.        <property name="suffix" value=".jsp"/>
29.    </bean>
30. </beans>
```

web.xml

```
1.    <?xml version="1.0" encoding="UTF-8"?>
2.    <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
3.        <display-name>app7</display-name>
4.        <welcome-file-list>
5.            <welcome-file>index.html</welcome-file>
6.            <welcome-file>index.htm</welcome-file>
7.            <welcome-file>index.jsp</welcome-file>
8.            <welcome-file>default.html</welcome-file>
9.            <welcome-file>default.htm</welcome-file>
10.           <welcome-file>default.jsp</welcome-file>
11.        </welcome-file-list>
12.        <servlet>
13.            <servlet-name>ds</servlet-name>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14. <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15. <load-on-startup>1</load-on-startup>
16. </servlet>
17. <servlet-mapping>
18. <servlet-name>ds</servlet-name>
19. <url-pattern>/</url-pattern>
20. </servlet-mapping>
21.</web-app>
```

5. AbstractWizardFormController:

In general, In web applications , it is required to display more than one page in order to complete a given task.

In web applications, providing sequence of pages in a single task is called as "Wizard".

IN general, in Web applications, we will use multiple pages to enter user details like general details, qualification details, communication details,...in User registration process, to achieve this we have to use "Wizard".

To support for the Wizards in web applications, Spring Web MVC has provided a predefined controller in the form of "org.springframework.web.servlet.mvc.AbstractWizardFormController".

AbstractWizardFormController allows us to carry the same command object through an entire flow of Web pages in Wizard.

If we want to use AbstractWizardController class in Spring web MVC applications then we have to use the following steps.

1. Prepare multiple web pages with the buttons where names of the buttons must be
 - a) _finish: Finish the wizard form.
 - b) _cancel: Cancel the wizard form.
 - c) _targetx: Move to the target page, where x is the zero-based page index.
2. Prepare Controller class by extending AbstractWizardFormController class and override the following methods.
 - a)protected ModelAndView processFinish(HttpServletRequest request, HttpServletResponse response, Object command, BindException exception) throws Exception
 - b)protected ModelAndView processCancel(HttpServletRequest request, HttpServletResponse response, Object command, BindException errors) throws Exception

Where processFinish() method will handle Wizard Finish action.

Where processCancel() method will handle Wizard cancel Task.

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**

 US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. Configure controller class configuration in Spring configuration file with "pages" property , where "pages" property must be of "list" type, it must include all the pages names in a sequence inorder to open the pages when we click on Next buttons in Wizard.

EX:

```
<bean name="/reg" class="com.durgasoft.controller.UserController">
  <property name="pages">
    <list>
      <value>form1</value>
      <value>form2</value>
      <value>form3</value>
    </list>
  </property>
</beans>
```

Example:

index.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br><br>
11. <jsp:forward page="welcomepage"/>
12. </body>
13. </html>
```

welcomepage.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h1 align="center">
11. <a href="reg">User Registration Form</a>
12. </h1>
13. </body>
14. </html>
```

**form1.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
4. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
12. <h3 style="color: blue;" align="center">User Registration Page</h3>
13. <form:form method="POST" commandName="user">
14. <center>
15. <table>
16. <tr>
17.   <td>First Name</td>
18.   <td><form:input path="fname"/>
19. </tr>
20. <tr>
21.   <td>Last Name</td>
22.   <td><form:input path="lname"/>
23. </tr>
24. <tr>
25.   <td colspan="2">
26.     <input type="submit" name="_target1" value="Next"/>
27.     <input type="submit" name="_cancel" value="Cancel"/>
28.   </td>
29. </tr>
30. </table>
31. </center>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
32. </form>
33. </body>
34. </html>
```

form2.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
4. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
12. <h3 style="color: blue;" align="center">User Registration Page</h3>
13. <form:form method="POST" commandName="user">
14. <center>
15. <table>
16. <tr>
17.   <td>User Qualification</td>
18.   <td><form:input path="uqual"/>
19. </tr>
20. <tr>
21.   <td>User Designation</td>
22.   <td><form:input path="udes"/>
23. </tr>
24. <tr>
25.   <td colspan="2">
26.     <input type="submit" name="_target0" value="Previous"/>
27.     <input type="submit" name="_target2" value="Next"/>
28.     <input type="submit" name="_cancel" value="Cancel"/>
29.   </td>
30. </tr>
31. </table>
32. </center>
33. </form:form>
34. </body>
35. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

form3.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
4. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body>
11. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
12. <h3 style="color: blue;" align="center">User Registration Page</h3>
13. <form:form method="POST" commandName="user">
14. <center>
15. <table>
16. <tr>
17.   <td>User Email Id</td>
18.   <td><form:input path="uemail"/>
19. </tr>
20. <tr>
21.   <td>User Mobile No</td>
22.   <td><form:input path="umobile"/>
23. </tr>
24. <tr>
25.   <td colspan="2">
26.     <input type="submit" name="_target1" value="Previous"/>
27.     <input type="submit" name="_finish" value="Finish"/>
28.     <input type="submit" name="_cancel" value="Cancel"/>
29.   </td>
30. </tr>
31. </table>
32. </center>
33. </form:form>
34. </body>
35. </html>
```

userdetails.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
```

CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28 **US NUM:** 4433326786**Mail ID:** durgasoftonlinetraining@gmail.com**WEBSITE:** www.durgasoftonline.com**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
3. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br><br>
11. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
12. <h3 style="color: blue;" align="center">User Registration Details</h3>
13. <center>
14. <table border="1">
15. <tr>
16.   <td>First Name</td>
17.   <td>${user.fname}</td>
18. </tr>
19. <tr>
20.   <td>Last Name</td>
21.   <td>${user.lname}</td>
22. </tr>
23. <tr>
24.   <td>User Qualification</td>
25.   <td>${user.uqual}</td>
26. </tr>
27. <tr>
28.   <td>User Designation</td>
29.   <td>${user.udes}</td>
30. </tr>
31. <tr>
32.   <td>User Email Id</td>
33.   <td>${user.uemail}</td>
34. </tr>
35. <tr>
36.   <td>User Mobile No</td>
37.   <td>${user.umobile}</td>
38. </tr>
39. </table>
40. <h3 align="center"></h3>
41. <a href="welcomepage">Welcome Page</a>
42. </center>
43. </body>
44. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

UserController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.validation.BindException;
7. import org.springframework.web.servlet.ModelAndView;
8. import org.springframework.web.servlet.mvc.AbstractWizardFormController;
9.
10. import com.durgasoft.command.User;
11.
12. public class UserController extends AbstractWizardFormController {
13.
14.     @Override
15.     protected ModelAndView processFinish(HttpServletRequest request, HttpServletResponse response, Object command,
16.                                         BindException exception) throws Exception {
17.         User user = (User)command;
18.         return new ModelAndView("userdetails", "user", user);
19.     }
20.
21.     @Override
22.     protected ModelAndView processCancel(HttpServletRequest request, HttpServletResponse response, Object command,
23.                                         BindException errors) throws Exception {
24.         // TODO Auto-generated method stub
25.         return new ModelAndView("welcomepage");
26.     }
27.
28.
29. }
```

User.java

```
1. package com.durgasoft.command;
2.
3. public class User {
4.     private String fname;
5.     private String lname;
6.     private String uqual;
7.     private String udes;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. private String uemail;
9. private String umobile;
10.
11. public String getFname() {
12.     return fname;
13. }
14. public void setFname(String fname) {
15.     this.fname = fname;
16. }
17. public String getLname() {
18.     return lname;
19. }
20. public void setLname(String lname) {
21.     this.lname = lname;
22. }
23. public String getUqual() {
24.     return uqual;
25. }
26. public void setUqual(String uqual) {
27.     this.uqual = uqual;
28. }
29. public String getUdes() {
30.     return udes;
31. }
32. public void setUdes(String udes) {
33.     this.udes = udes;
34. }
35. public String getUemail() {
36.     return uemail;
37. }
38. public void setUemail(String uemail) {
39.     this.uemail = uemail;
40. }
41. public String getUmobile() {
42.     return umobile;
43. }
44. public void setUmobile(String umobile) {
45.     this.umobile = umobile;
46. }
47.
48.
49.}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftware.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.   <bean name="/welcomepage" class="org.springframework.web.servlet.mvc.ParameterizableViewController">
12.     <property name="viewName" value="welcomepage"/>
13.   </bean>
14.   <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
15.   <bean name="/reg" class="com.durgasoft.controller.UserController">
16.     <property name="pages">
17.       <list>
18.         <value>form1</value>
19.         <value>form2</value>
20.         <value>form3</value>
21.       </list>
22.     </property>
23.     <property name="commandName" value="user"/>
24.     <property name="commandClass" value="com.durgasoft.command.User"/>
25.
26.   </bean>
27.   <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
28.     <property name="prefix" value="/WEB-INF/"/>
29.     <property name="suffix" value=".jsp"/>
30.   </bean>
31. </beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
   version="3.1">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3. <display-name>abstractwizardcontrollerapp</display-name>
4. <welcome-file-list>
5. <welcome-file>index.html</welcome-file>
6. <welcome-file>index.htm</welcome-file>
7. <welcome-file>index.jsp</welcome-file>
8. <welcome-file>default.html</welcome-file>
9. <welcome-file>default.htm</welcome-file>
10. <welcome-file>default.jsp</welcome-file>
11. </welcome-file-list>
12. <servlet>
13. <servlet-name>ds</servlet-name>
14. <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15. <load-on-startup>1</load-on-startup>
16. </servlet>
17. <servlet-mapping>
18. <servlet-name>ds</servlet-name>
19. <url-pattern>/</url-pattern>
20. </servlet-mapping>
21. </web-app>
```

The logo consists of the word "DURGATRY" written in a large, stylized, green font. The letters are slightly curved and overlap each other, creating a dynamic and modern appearance.**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Data Validations

The process of checking whether the data is valid or not before using data in application Logic is called as Data Validations.

There are two types of Data Validations.

1. Client Side Data Validations
2. Server Side Data Validations

1. Client Side Data Validations:

If we check whether data is valid or not at client browser after entering data in user forms and before submitting form to Server then it is called as Client Side Data Validations.

To perform Client Side data Validations we will use Java Script functions.

2. Server Side Data Validations:

If we check whether the data is valid or not after storing data at server side and before using data in application logic then it is called as Server side Data validations.

To perform Server side data validation we will use JAVA code at server side.

Spring is providing Server side Data validations in the following two ways.

1. By Using Spring provided Error class.
2. By Using Java provided validation annotations

1. Spring Data Validations By using Spring provided Error class:

To provide data validations in this approach we have to use the following approach.

1. Prepare Validator class by implementing "org.springframework.validation.Validator" interface.

Validator interface contains the following two methods.

- a) public boolean supports(Class<?> cls)

It will check whether the command class is right class or not for data validations on the basis of Class type.

- b) public void validate(Object command, Errors errors)

It will include Data Validation logic .

If any Validation error is identified then we have to use rejectValue() method

- b) public void rejectValue(String var_Name, String error_Code)

- c) Where variable name is a variable in Command class.

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- d) Where error_Code is either validation message directly or a key from the propertirs file which had validationr message.
2. Provide Properties file under src or in a package under properties file with messages in the form of key-value pairs.
 3. Configure Validator class and Properties file with "messageSource" property under org.springframework.context.support.ResourceBundleMessageSource
 4. Display Validation messages in User forms by using <form:errors/> tag.

Example:**index.jsp**

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="reg"/>
11. </body>
12. </html>
```

registrationform.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
4. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. <style type="text/css">
10. .error{
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11. color: red;
12. font-family: consolas;
13. font-style: italic;
14. font-weight: bold;
15. font-size: large;
16.}
17.</style>
18.</head>
19.<body>
20.<br><br><br>
21.<h2 style="color: red;" align="center">Durga Software Solutions</h2>
22.<h3 style="color: red;" align="center">Student Registration Form</h3>
23.<form:form method="POST" action="reg" commandName="user">
24.<center>
25.<table>
26.<tr>
27. <td>User Name</td>
28. <td><form:input path="uname"/></td>
29. <td><form:errors path="uname" cssClass="error"/>
30.</tr>
31.<tr>
32. <td>User Password</td>
33. <td><form:password path="upwd"/></td>
34. <td><form:errors path="upwd" cssClass="error"/>
35.</tr>
36.<tr>
37. <td>User Age</td>
38. <td><form:input path="uage"/></td>
39. <td><form:errors path="uage" cssClass="error"/>
40.</tr>
41.<tr>
42. <td>User Email Id</td>
43. <td><form:input path="uemail"/></td>
44. <td><form:errors path="uemail" cssClass="error"/>
45.</tr>
46.<tr>
47. <td>User Mobile No</td>
48. <td><form:input path="umobile"/></td>
49. <td><form:errors path="umobile" cssClass="error"/>
50.</tr>
51.<tr>
52. <td><input type="submit" value="Registration"/></td>
53.</tr>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

Mail ID: durgasoftonlinetraining@gmail.com +91- 7207 21 24 27/28WEBSITE: www.durgasoftonline.com

US NUM: 4433326786

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
54. </table>
55. </center>
56. </form:form>
57. </body>
58. </html>
```

registrationdetails.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br><br>
11. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
12. <h3 style="color: red;" align="center">User Registration Details</h3>
13. <center>
14. <table border="1">
15. <tr>
16.   <td>User Name</td>
17.   <td>${user.uname}</td>
18. </tr>
19. <tr>
20.   <td>User Password</td>
21.   <td>${user.upwd}</td>
22. </tr>
23. <tr>
24.   <td>User Age</td>
25.   <td>${user.uage}</td>
26. </tr>
27. <tr>
28.   <td>User Email Id</td>
29.   <td>${user.uemail}</td>
30. </tr>
31. <tr>
32.   <td>User Mobile No</td>
33.   <td>${user.umobile}</td>
34. </tr>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
35. </table>
36. </center>
37. </body>
38. </html>
```

User.java

```
1. package com.durgasoft.command;
2.
3. public class User {
4.     private String uname;
5.     private String upwd;
6.     private int uage;
7.     private String uemail;
8.     private String umobile;
9.
10.    public String getUsername() {
11.        return uname;
12.    }
13.    public void setUsername(String uname) {
14.        this.uname = uname;
15.    }
16.    public String getPassword() {
17.        return upwd;
18.    }
19.    public void setPassword(String upwd) {
20.        this.upwd = upwd;
21.    }
22.    public int getUsage() {
23.        return uage;
24.    }
25.    public void setUsage(int uage) {
26.        this.uage = uage;
27.    }
28.    public String getEmail() {
29.        return uemail;
30.    }
31.    public void setEmail(String uemail) {
32.        this.uemail = uemail;
33.    }
34.    public String getMobile() {
35.        return umobile;
36.    }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
37. public void setUmobile(String umobile) {  
38.     this.umobile = umobile;  
39. }  
40.  
41.  
42.}
```

UserController.java

```
1. package com.durgasoft.controller;  
2.  
3. import javax.servlet.http.HttpServletRequest;  
4. import javax.servlet.http.HttpServletResponse;  
5.  
6. import org.springframework.validation.BindException;  
7. import org.springframework.web.servlet.ModelAndView;  
8. import org.springframework.web.servlet.mvc.SimpleFormController;  
9.  
10. import com.durgasoft.command.User;  
11.  
12. public class UserController extends SimpleFormController {  
13.     @Override  
14.     protected ModelAndView onSubmit(HttpServletRequest request, HttpServletResponse response, Object command,  
15.                                         BindException errors) throws Exception {  
16.         User user = (User)command;  
17.         return new ModelAndView("registrationdetails", "user", user);  
18.     }  
19. }
```


UserValidator.java

```
1. package com.durgasoft.validator;  
2.  
3. import org.springframework.validation.Errors;  
4. import org.springframework.validation.ValidationUtils;  
5. import org.springframework.validation.Validator;  
6.  
7. import com.durgasoft.command.User;  
8.  
9. public class UserValidator implements Validator {  
10.  
11.     @Override
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12. public boolean supports(Class<?> cls) {
13.
14.     return User.class.isAssignableFrom(cls);
15. }
16.
17. @Override
18. public void validate(Object command, Errors errors) {
19.     ValidationUtils.rejectIfEmpty(errors, "uname", "uname.required");
20.     ValidationUtils.rejectIfEmpty(errors, "upwd", "upwd.required");
21.     ValidationUtils.rejectIfEmpty(errors, "uage", "uage.required");
22.     ValidationUtils.rejectIfEmpty(errors, "uemail", "uemail.required");
23.     ValidationUtils.rejectIfEmpty(errors, "umobile", "umobile.required");
24.
25.     User user = (User)command;
26.     if(!user.getUpwd().equals("") && user.getUpwd().length() < 6) {
27.         errors.rejectValue("upwd", "upwd.minLength");
28.     }
29.     if(!user.getUpwd().equals("") && user.getUpwd().length() > 10) {
30.         errors.rejectValue("upwd", "upwd.maxLength");
31.     }
32.     if(user.getUage() < 18 || user.getUage() > 30) {
33.         errors.rejectValue("uage", "uage.range");
34.     }
35.     if(!user.getUemail().equals("") && !user.getUemail().contains("@")) {
36.         errors.rejectValue("uemail", "uemail.invalid");
37.     }
38.     if(!user.getUmobile().equals("") && !user.getUmobile().startsWith("91-")) {
39.         errors.rejectValue("umobile", "umobile.invalid");
40.     }
41.
42. }
43.
44. }
```

validation_Messages.properties

- ✚ uname.required = User Name is Required.
- ✚ upwd.required = User Password is required.
- ✚ uage.required = User Age is Required.
- ✚ uemail.required = User Email Id Required.
- ✚ umobile.required = User Mobile Number is Required.
- ✚ upwd.minLength = User Password must be minimum 6 characters.
- ✚ upwd.maxLength = User Password must be maximum 10 characters.
- ✚ uage.range = User Age must be in the range from 18 to 30.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✚ uemail.invalid = User Email Id is Invalid.
- ✚ umobile.invalid = User Mobile Number is Invalid.

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.  <bean name="messageSource" class="org.springframework.context.support.ResourceB
  undleMessageSource">
13.    <property name="basename" value="com/durgasoft/resources/validation_Messages"/
  >
14.  </bean>
15.  <bean name="userValidator" class="com.durgasoft.validator.UserValidator"/>
16.  <bean name="/reg" class="com.durgasoft.controller.UserController">
17.    <property name="formView" value="registrationform"/>
18.    <property name="validator" ref="userValidator"/>
19.    <property name="commandName" value="user"/>
20.    <property name="commandClass" value="com.durgasoft.command.User"/>
21.  </bean>
22.
23.  <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanN
  ameUrlHandlerMapping"/>
24.
25. <bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceVi
  ewResolver">
26.   <property name="prefix" value="/WEB-INF/"/>
27.   <property name="suffix" value=".jsp"/>
28. </bean>
29. </beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app7</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```

2. Java Provided Validation API:

Validation-API is a specification provided JAVA and it is implemented by Third party vendors.

Hibernate has provided its validation API which contains Java provided Validation API and some extra validation rules.

1. @NotEmpty

- It will be applied to the bean variables or getter methods.
- It will check whether data is existed in the respective bean property or not.
- If the data is not existed in the bean property then it will raise validation error.
- It is applicable for only String type properties.

2. @NotNull

- It will be applied to the bean variables or getter methods.
- It will check whether data/value is existed in the respective bean property or not.
- If the data is not existed in the bean property then it will raise validation error.
- It is applicable for only Numeric Data type properties.

3. @Size

- It will be applied to the bean variables or getter methods.
- It will be applied for String type properties.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- It will check whether the properties value is as per the specified minimum length constraint and maximum length constraint or not.
- It will take two properties min and max.

Note: If we dont want to use @Size annotation then we will use @Min and @Max annotation

4. @Range

- It will be applied to the bean variables or getter methods.
- It is applied for the numeric data type properties.
- It will check the value as per the specified range.
- It will take two parameters min and max

5. @Pattern

- It will be applied either for property or for getter method directly.
- It will check whether the property data is as per the provided reg exp.
- It will take a parameter like "regexp", it will take pattern.

6. @Email

- It will be applied either for property or for getter method directly.
- It will check whether the value of a property is email id or not.
- It will not require any parameter.

7. @DateTimeFormat

- It will be applied either for property or for getter method directly.
- It will be applied for Date type properties.
- It will check whether the date value in a property is as per the specified format or not.
- It will take a parameter like "pattern" to represent Date type patterns.
- It will be applied for the properties in bean of Date type.

8. @Past

- It will be applied either for property or for getter method directly.
- It will check whether the date is past date or not.
- It will be applied for the properties in bean of Date type.

Steps:

1. Use all the validation related annotations in Bean classes.
2. Declare properties file under src and provide validation messages.
3. Enable Annotations in Spring application through Spring configuration file.
4. Display Validation messages inside the pages.

There are two ways to define validation messages in Spring applications.

1. Provide validation messages directly in the Annotation.

```
public class User{
    @NotEmpty(message="User Name is Required")
    private String uname
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
@Size(min=6, max=10, message="User password must be minimum 6 characters and  
maximum 10 characters)  
private String upwd;  
}
```

2. Provide validation messages through properties file.

validation_Messages.properties

NotEmpty.user.uname = User Name is Required.

Size.user.upwd = User Password must be minimum {2} characters and maximum {1} characters

To define validation messages in properties file then we have to use the following format.
Annotation_Name.Model_name.Prop_Name = Validation Message

EX:

NotEmpty.user.uname=User Name is Required.

NotEmpty.user.upwd=User Password is Required.

To enable Annotations in Spring applications we have to use the following tags in ds-servlet.xml

```
<context:component-scan base-package="com.durgasoft"/>  
<mvc:annotation-driven>
```

To declare controller class and to define request mapping for the Controller class we have to use the following annotations:

```
@Controller  
@RequestMapping
```

To use the above annotations in Spring web applications we have to use the following JARs in web application lib folder.

- 1) All Spring web MVC JARs
- 2) clasmate-version.jar
- 3) hibernate-validator-5.x/4.x.final.jar
- 4) jboss-logging-version.jar
- 5) validation-api-version.jar

Example:

index.jsp

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10.<jsp:forward page="reg"/>
11.</body>
12.</html>
```

registrationform.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
4. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. <style type="text/css">
10..error{
11.   color: red;
12.   font-family: consolas;
13.   font-style: italic;
14.   font-weight: bold;
15.   font-size: large;
16.}
17.</style>
18.</head>
19.<body>
20.<br><br><br>
21.<h2 style="color: red;" align="center">Durga Software Solutions</h2>
22.<h3 style="color: red;" align="center">Student Registration Form</h3>
23.<form:form method="POST" action="reg" commandName="user">
24.<center>
25.<table>
26.<tr>
27.   <td>User Name</td>
28.   <td><form:input path="uname"/></td>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
29. <td><form:errors path="uname" cssClass="error"/>
30.</tr>
31.<tr>
32. <td>User Password</td>
33. <td><form:password path="upwd"/></td>
34. <td><form:errors path="upwd" cssClass="error"/>
35.</tr>
36.<tr>
37. <td>User Age</td>
38. <td><form:input path="uage"/></td>
39. <td><form:errors path="uage" cssClass="error"/>
40.</tr>
41.<tr>
42. <td>User Date Of Birth</td>
43. <td><form:input path="udob"/></td>
44. <td><form:errors path="udob" cssClass="error"/>
45.</tr>
46.<tr>
47. <td>User Email Id</td>
48. <td><form:input path="uemail"/></td>
49. <td><form:errors path="uemail" cssClass="error"/>
50.</tr>
51.<tr>
52. <td>User Mobile No</td>
53. <td><form:input path="umobile"/></td>
54. <td><form:errors path="umobile" cssClass="error"/>
55.</tr>
56.<tr>
57. <td><input type="submit" value="Registration"/></td>
58.</tr>
59.</table>
60.</center>
61.</form:form>
62.</body>
63. </html>
```

registrationdetails.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br><br>
11. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
12. <h3 style="color: red;" align="center">User Registration Details</h3>
13. <center>
14. <table border="1">
15. <tr>
16.   <td>User Name</td>
17.   <td>${user.uname}</td>
18. </tr>
19. <tr>
20.   <td>User Password</td>
21.   <td>${user.upwd}</td>
22. </tr>
23.
24. <tr>
25.   <td>User Age</td>
26.   <td>${user.uage}</td>
27. </tr>
28. <tr>
29.   <td>User Date Of Birth</td>
30.   <td>${user.udob}</td>
31. </tr>
32. <tr>
33.   <td>User Email Id</td>
34.   <td>${user.uemail}</td>
35. </tr>
36. <tr>
37.   <td>User Mobile No</td>
38.   <td>${user.umobile}</td>
39. </tr>
40. </table>
41. </center>
42. </body>
43. </html>
```

User.java

```
1. package com.durgasoft.command;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.  
3. import java.util.Date;  
4.  
5.  
6. import javax.validation.constraints.NotNull;  
7. import javax.validation.constraints.Past;  
8. import javax.validation.constraints.Pattern;  
9. import javax.validation.constraints.Size;  
10.  
11. import org.hibernate.validator.constraints.Email;  
12. import org.hibernate.validator.constraints.NotEmpty;  
13. import org.hibernate.validator.constraints.Range;  
14. import org.springframework.format.annotation.DateTimeFormat;  
15.  
16.  
17. public class User {  
18.     @NotEmpty  
19.     private String uname;  
20.  
21.     @NotEmpty  
22.     @Size(max=10, min=6)  
23.     private String upwd;  
24.  
25.     @NotNull  
26.     @Range(min=18, max=20)  
27.     private int uage;  
28.  
29.     @NotNull  
30.     @DateTimeFormat(pattern="dd/MM/yyyy")  
31.     @Past  
32.     private Date udob;  
33.  
34.     @NotEmpty  
35.     @Email  
36.     private String uemail;  
37.  
38.     @NotEmpty  
39.     @Pattern(regexp="91-[0-9]{10}")  
40.     private String umobile;  
41.  
42.     public String getUname() {  
43.         return uname;  
44.     }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
45. public void setUserName(String uname) {  
46.     this.uname = uname;  
47. }  
48. public String getUpwd() {  
49.     return upwd;  
50. }  
51. public void setUpwd(String upwd) {  
52.     this.upwd = upwd;  
53. }  
54. public int getUage() {  
55.     return uage;  
56. }  
57. public void setUage(int uage) {  
58.     this.uage = uage;  
59. }  
60. public void setUdob(Date udob) {  
61.     this.udob = udob;  
62. }  
63. public Date getUdob() {  
64.     return udob;  
65. }  
66. public String getUemail() {  
67.     return uemail;  
68. }  
69. public void setUemail(String uemail) {  
70.     this.uemail = uemail;  
71. }  
72. public String getUmobile() {  
73.     return umobile;  
74. }  
75. public void setUmobile(String umobile) {  
76.     this.umobile = umobile;  
77. }  
78.  
79.  
80.}
```

UserController.java

```
1. package com.durgasoft.controller;  
2.  
3.  
4. import javax.validation.Valid;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5.  
6. import org.springframework.stereotype.Controller;  
7. import org.springframework.ui.Model;  
8. import org.springframework.validation.BindingResult;  
9. import org.springframework.web.bind.annotation.RequestMapping;  
10. import org.springframework.web.bind.annotation.RequestMethod;  
11. import org.springframework.web.servlet.ModelAndView;  
12. import org.springframework.web.servlet.config.annotation.EnableWebMvc;  
13.  
14. import com.durgasoft.command.User;  
15.  
16. @Controller  
17. @RequestMapping("/reg")  
18.  
19. public class UserController {  
20.     @RequestMapping(method = RequestMethod.GET)  
21.     public String showPage(Model model) {  
22.         model.addAttribute("user", new User());  
23.         return "registrationform";  
24.     }  
25.  
26.     @RequestMapping(method = RequestMethod.POST)  
27.     public ModelAndView registration(@Valid User user, BindingResult errors, Model model)  
    ) {  
28.         model.addAttribute("user", user);  
29.         if(errors.hasErrors()) {  
30.             return new ModelAndView("registrationform", "user", user);  
31.         } else {  
32.             return new ModelAndView("registrationdetails", "user", user);  
33.         }  
34.     }  
35. }
```

validation_Messages.properties

- ✚ typeMismatch = {0} is in invalid format
- ✚ NotEmpty.user.uname = User Name is Required.
- ✚ NotEmpty.user.upwd = User Password is Required.
- ✚ Size.user.upwd = User Password must be minimum {2} characters and maximum {1} characters.
- ✚ NotNull.user.uage = User Age is Required.
- ✚ Range.user.uage = User Age must be in the range from {2} through {1} Years.
- ✚ NotNull.user.udob = User Date Of Birth is Required.
- ✚ DateTimeFormat.user.udob = User Date Of Birth is Invalid.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- Past.user.udob = User Date of Birth Must be Past Date.
- NotEmpty.user.uemail = User Email Id is Required.
- Email.user.uemail = User Email Id is Invalid.
- NotEmpty.user.umobile = User Mobile No is Required.
- Pattern.user.umobile = User Mobile No must be in the pattern like "91-DDDDDDDDDD".

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns = "http://www.springframework.org/schema/beans"
3.   xmlns:context = "http://www.springframework.org/schema/context"
4.   xmlns:mvc = "http://www.springframework.org/schema/mvc"
5.   xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
6.   xsi:schemaLocation =
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context-3.0.xsd
11.    http://www.springframework.org/schema/mvc
12.    http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd">
13.
14.  <context:component-scan base-package="com.durgasoft"/>
15.  <mvc:annotation-driven/>
16.
17.  <bean name="messageSource" class="org.springframework.context.support.ResourceB
  undleMessageSource">
18.    <property name="basename" value="com/durgasoft/resources/validation_Messages"/
  >
19.  </bean>
20.
21.  <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanN
  ameUrlHandlerMapping"/>
22.
23. <bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceVi
  ewResolver">
24. <property name="prefix" value="/WEB-INF/"/>
25. <property name="suffix" value=".jsp"/>
26. </bean>
27. </beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app7</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```

Extra jars required along with Spring jars:

- ⊕ classmate-0.8.0.jar
- ⊕ hibernate-validator-5.0.1.final.jar
- ⊕ jboss-logging-3.1.0.ga.jar
- ⊕ validation-api-1.1.0.final.jar

Example on Data Validations through Java Validation API provided Annotations by using properties file provided validation Messages:

index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="reg"/>
11. </body>
12. </html>
```

reg_form.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
4. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. <style type="text/css">
10. .error{
11.   color: red;
12.   font-family: consolas;
13.   font-style: italic;
14.   font-weight: bold;
15.   font-size: large;
16. }
17. </style>
18. </head>
19. <body>
20. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
21. <h3 style="color: blue;" align="center">User Registration Form</h3>
22. <form:form method="POST" action="reg" commandName="user">
23. <center>
24. <table>
25. <tr>
26.   <td>User Name</td>
27.   <td><form:input path="uname"/></td>
28.   <td><form:errors path="uname" cssClass="error"/></td>
29. </tr>
30. <tr>
31.   <td>User Password</td>
32.   <td><form:password path="upwd"/></td>
33.   <td><form:errors path="upwd" cssClass="error"/></td>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
34. </tr>
35. <tr>
36.   <td>User Age</td>
37.   <td><form:input path="uage"/></td>
38.   <td><form:errors path="uage" cssClass="error"/></td>
39. </tr>
40. <tr>
41.   <td>User Date Of Birth</td>
42.   <td><form:input path="udob"/></td>
43.   <td><form:errors path="udob" cssClass="error"/></td>
44. </tr>
45. <tr>
46.   <td>User Email Id</td>
47.   <td><form:input path="uemail"/></td>
48.   <td><form:errors path="uemail" cssClass="error"/></td>
49. </tr>
50. <tr>
51.   <td>User Mobile No</td>
52.   <td><form:input path="umobile"/></td>
53.   <td><form:errors path="umobile" cssClass="error"/></td>
54. </tr>
55. <tr>
56.   <td><input type="submit" value="Registration"/></td>
57. </tr>
58. </table>
59. </center>
60. </form:form>
61. </body>
62. </html>
```

registrationdetails.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

Mail ID: durgasoftonlinetraining@gmail.com

+91- 7207 21 24 27/28

WEBSITE: www.durgasoftonline.com

US NUM: 4433326786

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11. <h3 style="color: blue;" align="center">User Registration Form</h3>
12. <center>
13. <table border="1">
14. <tr>
15.   <td>User Name</td>
16.   <td>${user.uname}</td>
17. </tr>
18. <tr>
19.   <td>User Password</td>
20.   <td>${user.upwd}</td>
21. </tr>
22. <tr>
23.   <td>User Age</td>
24.   <td>${user.uage}</td>
25. </tr>
26. <tr>
27.   <td>User Date Of Birth</td>
28.   <td>${user.udob}</td>
29. </tr>
30. <tr>
31.   <td>User Email Id</td>
32.   <td>${user.uemail}</td>
33. </tr>
34. <tr>
35.   <td>User Mobile No</td>
36.   <td>${user.umobile}</td>
37. </tr>
38. </table>
39. </center>
40. </body>
41. </html>
```

User.java

```
1. package com.durgasoft.command;
2.
3. import java.util.Date;
4.
5. import javax.validation.constraints.NotNull;
6. import javax.validation.constraints.Past;
7. import javax.validation.constraints.Pattern;
8. import javax.validation.constraints.Size;
9.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10. import org.hibernate.validator.constraints.Email;
11. import org.hibernate.validator.constraints.NotEmpty;
12. import org.hibernate.validator.constraints.Range;
13. import org.springframework.format.annotation.DateTimeFormat;
14.
15. public class User {
16.     @NotEmpty(message="User Name is required")
17.     private String uname;
18.
19.     @NotEmpty(message="User Password is Required")
20.     @Size(min=6, max=10, message="User Password must be minimum 6 and maximum 10 characters")
21.     private String upwd;
22.
23.     @NotNull(message="User is Required")
24.     @Range(min=18, max=25, message="User Age must be minimum 18 and maximum 25")
25. }
25. private int uage;
26.
27. @NotNull(message="User Date of Birth is Required")
28. @DateTimeFormat(pattern="dd/MM/yyyy")
29. @Past(message="User DOB must be past date")
30. private Date udob;
31.
32. @NotEmpty(message="User Email Id is Required")
33. @Email
34. private String uemail;
35.
36. @NotEmpty(message="User Mobile No is Required")
37. @Pattern(regexp="91-[0-9]{10}", message="User Mobile Number must be in 91-DDDDDDDDDD format")
38. private String umobile;
39.
40. public String getUname() {
41.     return uname;
42. }
43. public void setUname(String uname) {
44.     this.uname = uname;
45. }
46. public String getUpwd() {
47.     return upwd;
48. }
49. public void setUpwd(String upwd) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
50.     this.upwd = upwd;
51. }
52. public int getUage() {
53.     return uage;
54. }
55. public void setUage(int uage) {
56.     this.uage = uage;
57. }
58. public Date getUdob() {
59.     return udob;
60. }
61. public void setUdob(Date udob) {
62.     this.udob = udob;
63. }
64. public String getUemail() {
65.     return uemail;
66. }
67. public void setUemail(String uemail) {
68.     this.uemail = uemail;
69. }
70. public String getUmobile() {
71.     return umobile;
72. }
73. public void setUmobile(String umobile) {
74.     this.umobile = umobile;
75. }
76.
77.
78.}
```

UserController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.validation.Valid;
4.
5. import org.springframework.stereotype.Controller;
6. import org.springframework.ui.Model;
7. import org.springframework.validation.BindingResult;
8. import org.springframework.web.bind.annotation.RequestMapping;
9. import org.springframework.web.bind.annotation.RequestMethod;
10. import org.springframework.web.servlet.ModelAndView;
11.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12. import com.durgasoft.command.User;
13.
14. @Controller
15. @RequestMapping("/reg")
16. public class UserController {
17.     @RequestMapping(method = RequestMethod.GET)
18.     public String showForm(Model model) {
19.         model.addAttribute("user", new User());
20.         return "reg_form";
21.     }
22.
23.     @RequestMapping(method = RequestMethod.POST)
24.     public ModelAndView registration(@Valid User user, BindingResult errors, Model model
    ) {
25.         if(errors.hasErrors()) {
26.             return new ModelAndView("reg_form", "user", user);
27.         }else {
28.             return new ModelAndView("registrationdetails", "user", user);
29.         }
30.     }
31. }
```

error_Messages.properties

+ typeMismatch = {0} is in invalid format

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns = "http://www.springframework.org/schema/beans"
3.     xmlns:context = "http://www.springframework.org/schema/context"
4.     xmlns:mvc = "http://www.springframework.org/schema/mvc"
5.     xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
6.     xsi:schemaLocation =
7.         http://www.springframework.org/schema/beans
8.         http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
9.         http://www.springframework.org/schema/context
10.        http://www.springframework.org/schema/context/spring-context-3.0.xsd
11.        http://www.springframework.org/schema/mvc
12.        http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd">
13.        <context:component-scan base-package="com.durgasoft"/>
14.        <mvc:annotation-driven/>
15.        <bean name="messageSource" class="org.springframework.context.support.ResourceB
    undleMessageSource">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
16.   <property name="basename" value="com/durgasoft/resources/error_Messages"/>
17. </bean>
18. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
19.
20. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
21.   <property name="prefix" value="/WEB-INF/"/>
22.   <property name="suffix" value=".jsp"/>
23. </bean>
24.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app14</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

HandlerMappings

- HandlerMapping is a component in Spring Web MVC Framework, it will take the incoming request URI and it will identify the mapped Controller class through Spring configuration file and return that Controller class details to DispatcherServlet.
- Spring Web MVC has provided a set of predefined HandlerMappings in the package like "org.springframework.web.servlet.handler".
- Spring Web MVC has provided the following predefined HandlerMapping classes.
 - BeanNameUrlHandlerMapping
 - SimpleUrlHandlerMapping
 - ControllerClassNameHandlerMapping
 - DefaultAnnotationHandlerMapping

1. BeanNameUrlHandlerMapping:

- It is the default HandlerMapping in Spring web MVC applications if we use XML configurations, it is not required to configure in Spring configuration file.
- BeanNameUrlHandlerMapping is able to identify the Controller classes by mapping the incoming request URI with the Bean identity["id" attribute value] or Bean logical name["name" attribute value] which we defined in Spring configuration file.

EX:

ds-servlet.xml

```
<beans>
-----
<bean name="handlerMapping"
class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
<bean name="/welcome" class="com.durgasoft.controller.WelcomeController"/>
-----
</beans>
```

If we submit request with the URI "welcome" then BeanNameUrlHandlerMapping is able to map "welcome" with "name" attribute value in <bean> tag , if it is matched then BeanNameUrlHandlerMapping will confirm the respective Controller class is "com.durgasoft.controller.WelcomeController" and send its details to DispatcherServlet.

EX: <http://localhost:1010/app/welcome>

Example:

index.jsp

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.    pageEncoding="ISO-8859-1">%>
3.  <!DOCTYPE html PUBLIC "-
   /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4.  <html>
5.  <head>
6.  <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7.  <title>Insert title here</title>
8.  </head>
9.  <body>
10. <jsp:forward page="helloform"/>
11. </body>
12. </html>
```

helloform.jsp

```
1.  <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.  pageEncoding="ISO-8859-1"%>
3.  <!DOCTYPE html PUBLIC "-
   /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4.  <html>
5.  <head>
6.  <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7.  <title>Insert title here</title>
8.  </head>
9.  <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Form </h3>
12. <form method="POST" action="wish">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Status </h3>
12. <h1 style="color: red;" align="center"> Hello ${uname} </h1>
13. </body>
14. </html>
```

HelloFormController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6.
7. import org.springframework.web.servlet.ModelAndView;
8. import org.springframework.web.servlet.mvc.Controller;
9.
10. public class HelloFormController implements Controller {
11.
12.     @Override
13.     public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception {
14.         return new ModelAndView("helloform");
15.     }
16. }
```

WishController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
9. public class WishController implements Controller {
10.
11.     @Override
12.     public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception {
13.         String uname = request.getParameter("uname");
14.         return new ModelAndView("wish", "uname", uname);
15.     }
16. }
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:p="http://www.springframework.org/schema/p"
5.     xmlns:context="http://www.springframework.org/schema/context"
6.     xsi:schemaLocation="
7.         http://www.springframework.org/schema/beans
8.         http://www.springframework.org/schema/beans/spring-beans.xsd
9.         http://www.springframework.org/schema/context
10.        http://www.springframework.org/schema/context/spring-context.xsd">
11.
12.    <bean name="/helloform" class="com.durgasoft.controller.HelloFormController"/>
13.    <bean name="/wish" class="com.durgasoft.controller.WishController"/>
14.
15.    <!--
16.        <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/> -->
17.
18.    <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
19.        <property name="prefix" value="/WEB-INF/"/>
20.        <property name="suffix" value=".jsp"/>
21.    </bean>
22. </beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app17</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```

2. SimpleUrlHandlerMapping:

org.springframework.web.servlet.handler.SimpleUrlHandlerMapping is able to map the incoming request with a particular Controller class and it will provide the respective Controller class details to DispatcherServlet with the following actions.

- a) SimpleUrlHandlerMapping will take incoming request URI value.
- b) It will search for the Request URI name in its Properties which are configured in Spring Configuration file.
- c) If any property key is matched with incoming request URI value then SimpleUrlHandlerMapping will take the corresponding property value , that is, logical name of the Controller class.
- d) SimpleUrlHandlerMapping will identify the controller class on the basis of logical name which we get from the matched property.

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

EX:

```

1. <beans>
2. ----
3. <bean name="wishController" class="com.durgasoft.controller.WishController"/>
4. <bean name="welcomeController" class="com.durgasoft.controller.WelcomeController"/>

5. <bean name="handlerMapping" class="org....handler.SimpleUrlHandlerMapping">
6.   <property name="mappings">
7.     <props>
8.       <prop key="/wish">wishController</prop>
9.       <prop key="/welcome">welcomeController</prop>
10.      </props>
11.    </property>
12.  </bean>
13. ----
14. </beans>
```

From the above example, if we submit request with the URI "wish" then SimpleUrlHandlerMapping will map this URI with properties names configured in Spring configuration file and it will identify "wishController" as value and it will identify the respective controller class like "com.durgasoft.controllers.WishController" from Spring configuration file.

EX:

http://localhost:1010/app/wish ----> wishController ---> WishController
 http://localhost:1010/app/welcome ---> welcomeController --> WelcomeController

Example:

index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-
   /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="hello"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

11. </body>
12. </html>

helloform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red; align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue; align="center"> User Hello Form </h3>
12. <form method="POST" action="wish">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. </head>
9. <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Status </h3>
12. <h1 style="color: red;" align="center"> Hello ${uname} </h1>
13. </body>
14. </html>
```

**HelloFormController.java**

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6.
7. import org.springframework.web.servlet.ModelAndView;
8.
9. import org.springframework.web.servlet.mvc.Controller;
10.
11. public class HelloFormController implements Controller {
12.
13.     @Override
14.     public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse
e response) throws Exception {
15.         return new ModelAndView("helloform");
16.     }
17. }
```

WishController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
9. public class WishController implements Controller {
10.
11.     @Override
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12. public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception {
13.     String uname = request.getParameter("uname");
14.     return new ModelAndView("wish", "uname", uname);
15. }
16.

```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <bean name="helloFormController" class="com.durgasoft.controller.HelloFormController"/>
13. <bean name="wishController" class="com.durgasoft.controller.WishController"/>
14.
15. <!--
16.   <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/> -->
17. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
18.   <property name="mappings">
19.     <props>
20.       <prop key="/hello">helloFormController</prop>
21.       <prop key="/wish">wishController</prop>
22.     </props>
23.   </property>
24. </bean>
25. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
26.   <property name="prefix" value="/WEB-INF/"/>
27.   <property name="suffix" value=".jsp"/>
28. </bean>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app17</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21. </web-app>
```

3. ControllerClassNameHandlerMapping

org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping is able to map the incoming request URI with the Controller class by following the below convention.

- a) It will take request URI value and removes extension if any extension.
- b) It will make the first letter as Upper Case letter in the Request URI value.
- c) It will append Controller to the modified Request URI value.
- d) It will search for the respective controller class with the above convention and sends that Controller class details to DispatcherServlet.

If we submit a request with the URI value "wish" then ControllerClassNameHandlerMapping will search for the Controller class with the name "WishController", in this context, "name" attribute must not be provided in the Controller class configuration

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

EX:

```
<beans ...>
<bean name="handlerMapping"
class="org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping"/>
<bean class="com.durgasoft.controller.HelloController" />
<bean class="com.durgasoft.controller.WelcomeController" />
</beans>
```

In the above example, if we submit request with the Request URI "wish" then WishController class will be mapped.

If we submit request with the URI "welcome" then WelcomeController class will be mapped.

EX:

helloform ---->HelloFormController --> Valid
helloForm ---->HelloFormController --> Invalid, where 'F' is not matched.

If we want to map the request URI with the Controller class names in case insensitive manner then we have to use "caseSensitive" property with the value "true".

EX:

```
<beans ...>
<bean name="handlerMapping"
class="org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping">
<property name="caseSensitive" value="true"/>
<bean class="com.durgasoft.controller.HelloController" />
<bean class="com.durgasoft.controller.WelcomeController" />
</beans>
```

EX:

helloform ---->HelloFormController --> Invalid, 'f' is not matched.
helloForm ---->HelloFormController --> valid.

EXAMPLE:**index.jsp**

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="helloForm"/>
11. </body>
12. </html>
```

helloform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Form </h3>
12. <form method="POST" action="wish">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Status </h3>
12. <h1 style="color: red;" align="center"> Hello ${uname} </h1>
13. </body>
14. </html>
```

HelloFormController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6.
7. import org.springframework.web.servlet.ModelAndView;
8.
9. import org.springframework.web.servlet.mvc.Controller;
10. public class HelloFormController implements Controller {
11.
12.     @Override
13.     public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse response) throws Exception {
14.
15.         return new ModelAndView("helloform");
16.     }
17. }
```

WishController.java

```
1. package com.durgasoft.controller;
2.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.web.servlet.ModelAndView;
7. import org.springframework.web.servlet.mvc.Controller;
8.
9. public class WishController implements Controller {
10.
11.     @Override
12.     public ModelAndView handleRequest(HttpServletRequest request, HttpServletResponse) throws Exception {
13.         String uname = request.getParameter("uname");
14.         return new ModelAndView("wish", "uname", uname);
15.     }
16. }
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <bean class="com.durgasoft.controller.HelloFormController"/>
13. <bean class="com.durgasoft.controller.WishController"/>
14.
15. <!--
16.     <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/> -->
17. <!--
18.     <bean name="handlerMapping" class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
19.         <property name="mappings">
20.             <props>
21.                 <prop key="/hello">helloFormController</prop>
22.                 <prop key="/wish">wishController</prop>
23.             </props>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
23. </property>
24. </bean>
25. -->
26. <bean name="handlerMapping" class="org.springframework.web.servlet.mvc.support.C
ontrollerClassNameHandlerMapping">
27.   <property name="caseSensitive" value="true"/>
28. </bean>
29. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalReso
urceViewResolver">
30.   <property name="prefix" value="/WEB-INF/"/>
31.   <property name="suffix" value=".jsp"/>
32. </bean>
33.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app17</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.  </welcome-file-list>
12.  <servlet>
13.    <servlet-name>ds</servlet-name>
14.    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.    <load-on-startup>1</load-on-startup>
16.  </servlet>
17.  <servlet-mapping>
18.    <servlet-name>ds</servlet-name>
19.    <url-pattern>/</url-pattern>
20.  </servlet-mapping>
21.</web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4. DefaultAnnotationHandlerMapping:

org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping is a default HandlerMapping class in Spring WEB MVC applications if we use Annotations , it is not required to configure in Spring configuration File.

It will map incoming request URI with the name provided along with the @RequestMapping annotation which we provided above of the controller class or Controller class method.

EX:**index.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//
4.   /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <jsp:forward page="wish"/>
11. </body>
12. </html>
```

helloform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//
4.   /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Form </h3>
12. <form method="POST" action="wish">
13. <center>
```

CONTACT US:**Mobile: +91- 8885 25 26 27**

+91- 7207 21 24 27/28

**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```

14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

wish.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Status </h3>
12. <h1 style="color: red;" align="center"> Hello ${uname} </h1>
13. </body>
14. </html>
```

WishController.java

```

1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.stereotype.Controller;
7. import org.springframework.web.bind.annotation.RequestMapping;
8. import org.springframework.web.bind.annotation.RequestMethod;
9. import org.springframework.web.servlet.ModelAndView;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10. @Controller
11. @RequestMapping("/wish")
12. public class WishController{
13.     @RequestMapping(method=RequestMethod.GET)
14.     public String helloForm() {
15.         return "helloform";
16.     }
17.     @RequestMapping(method=RequestMethod.POST)
18.     public ModelAndView wish(HttpServletRequest request, HttpServletResponse response
    ) throws Exception {
19.         String uname = request.getParameter("uname");
20.         return new ModelAndView("wish", "uname", uname);
21.     }
22. }
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <bean class="com.durgasoft.controller.WishController"/>
13.
14. <!--
15. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.BeanNa
16. meUrlHandlerMapping"/> -->
17. <!--
18. <bean name="handlerMapping" class="org.springframework.web.servlet.handler.Simple
19. UrlHandlerMapping">
20.   <property name="mappings">
21.     <props>
22.       <prop key="/hello">helloFormController</prop>
23.       <prop key="/wish">wishController</prop>
24.     </props>
25.   </property>
26. </bean>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
24.  
25. <bean name="handlerMapping" class="org.springframework.web.servlet.mvc.support.C  
ontrollerClassNameHandlerMapping">  
26.   <property name="caseSensitive" value="true"/>  
27. </bean>  
28. -->  
29. <bean name="handlerMapping" class="org.springframework.web.servlet.mvc.annotation  
.DefaultAnnotationHandlerMapping"/>  
30.  
31. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalReso  
urceViewResolver">  
32.   <property name="prefix" value="/WEB-INF/"/>  
33.   <property name="suffix" value=".jsp"/>  
34. </bean>  
35.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>  
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-  
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.  
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-  
app_4_0.xsd" id="WebApp_ID" version="4.0">  
3. <display-name>app17</display-name>  
4. <welcome-file-list>  
5.   <welcome-file>index.html</welcome-file>  
6.   <welcome-file>index.htm</welcome-file>  
7.   <welcome-file>index.jsp</welcome-file>  
8.   <welcome-file>default.html</welcome-file>  
9.   <welcome-file>default.htm</welcome-file>  
10.  <welcome-file>default.jsp</welcome-file>  
11. </welcome-file-list>  
12. <servlet>  
13.   <servlet-name>ds</servlet-name>  
14.   <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  
15.   <load-on-startup>1</load-on-startup>  
16. </servlet>  
17. <servlet-mapping>  
18.   <servlet-name>ds</servlet-name>  
19.   <url-pattern>/</url-pattern>  
20. </servlet-mapping>  
21.</web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: In Single Spring WEB MVC applications we are able to configure more than one HandlerMapping classes in spring configuration file, if we want to provide a particular order to execute them then we have to declare "order" property with a particular value at each and every HandlerMapping class, where if the order value is low then the HandlerMapping class will get high priority and if the order value is high then the respective Handlermapping class will get less priority.

**EX:**

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans ...>
3.   <bean class="com.durgasoft.controller.WishController"/>
4.   <bean name="handlerMapping"           class="org.springframework.web.servlet.handle
r.BeanNameUrlHandlerMapping">
5.     <property name="order" value="1"/>
6.   </bean>
7.   <bean name="handlerMapping"           class="org.springframework.web.servlet.handler.Si
mpleUrlHandlerMapping">
8.     <property name="mappings">
9.       <props>
10.         <prop key="/hello">helloFormController</prop>
11.         <prop key="/wish">wishController</prop>
12.       </props>
13.     </property>
14.     <property name="order" value="2"/>
15.   </bean>
16.
17.   <bean name="handlerMapping" class="org.springframework.web.servlet.mvc.support.Co
ntrollerClassNameHandlerMapping">
18.     <property name="caseSensitive" value="true"/>
19.     <property name="order" value="3"/>
20.   </bean>
21.   <bean name="handlerMapping" class="org.springframework.web.servlet.mvc.annotation.
DefaultAnnotationHandlerMapping">
22.     <property name="order" value="4"/>
23.   </bean>
24.   <bean name="viewResolver"    class="org.springframework.web.servlet.view.InternalRe
sourceViewResolver">
25.     <property name="prefix" value="/WEB-INF/"/>
26.     <property name="suffix" value=".jsp"/>
27.   </bean>
28. </beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

HandlerInterceptor

In general, in web applications, to provide pre-processing and post-processing services for web resources like Servlets and JSPs we will use "Servlet Filters".

In web applications, we are able to provide single filter for multiple Servlets and JSPs and we are able to provide multiple filters for Single Servlet or JSP.

In the above representation, when we submit request to the container for a particular Servlet then container will perform the following actions.

1. Container will check whether any Filter or Filters are associated with the respective Servlet or not.
2. If any Filter or Filters are associated with the Servlet then container will execute all the Filters in one by one fashion in an order which we configured in web.xml file.
3. If the present request is satisfying Filter constraints then Container will forward request to the next filter or to the Servlet if no filter is existed in next.
4. If the present request is not satisfying the Filter constraints then Filter will send response to the client through all the previous filters.
5. If all the Filters are executed then Container will execute the respective Servlet, after execution of the Servlet Container will send the generated response to the client through all the Filters in reverse order.

Similarly in Spring Web MVC applications, if we want to provide Pre-Processing and Post-Processing services to any controller or all the controllers components then we have to use Interceptors.

If we provide Interceptors in Spring WEB MVC applications then they will be executed before executing the controller, after executing the controller and then after sending response to client.

If we provide Interceptors in Spring WEB MVC applications then they will be executed before executing the controller, after executing the controller and before sending response to client and then after sending response to client.

If we want to use Interceptors in Spring WEB MVC applications then we have to use the following two steps.

1. Prepare Interceptor class
2. Configure Interceptor class in Spring configuration File.

1. Prepare Interceptor class:

To prepare Interceptor class we have to declare an user defined class and either implement org.springframework.web.servlet.HandlerInterceptor interface or extend org.springframework.web.servlet.handler.HandlerInterceptorAdapter class in User defined class.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where HandlerInterceptor is an interface contains the following three methods.

1. public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler) throws Exception

- ✓ This method will include pre-processing logic which we want to execute before executing controller method
- ✓ This method will be executed before executing controller class.
- ✓ This method will return a boolean value like true or false.
- ✓ If this method returns 'true' value then DispatcherServlet will understand the present Interceptor is executed successfully and request will be forwarded to the next interceptor or to the Controller if no interceptor is existed.
- ✓ If this method returns 'false' value then the present Interceptor constraints are not satisfied by the present request and Interceptor generates the required response to the client through all the previous interceptors with out executing the respective Controller.

2. public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler,

ModelAndView modelAndView) throws Exception

- ✓ This method will include the Post-Processing services logic which we want to execute after executing the controller component.
- ✓ This method will be executed after executing the controller component and before submitting response to client.
- ✓ This method can be used to add additional attribute to the ModelAndView object to be used in the view pages.
- ✓ This method can be used to determine the time taken by handler method to process the client request.

3. public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler,

Exception e) throws Exception

- ✓ This method will be executed after executing the controller class and after sending response to client.
- ✓ This method will be executed only once for all the controller classes, that is, after the response rendering.

Note: In Spring WEB MVC applications, preHandler() and postHandle() methods are executed for each and every controller, but, afterCompletion() method will be executed only once after the response rendering.

If more than one Interceptor is provided for a controller component then all the Interceptors are executed like below.

If we provide Multiple Interceptors in Spring WEB MVC application then preHandle() methods will be executed as per the Interceptors configuration order which we configured in Spring

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

configuration file, but, postHandle() and afterCompletion() methods are executed in reverse order of the Interceptors.

2. Configure Interceptor class in Spring configuration File.

To configure Interceptor class in Spring configuration file , we have to use the following approaches.

1. Use "interceptors" property of type list in HandlerMapping configuration
2. User <interceptors> tag in spring configuration file directly.
3. Use <mvc:interceptors> tag in Spring configuration file.

1. Use "interceptors" property of type list in HandlerMapping configuration:

```
<beans>
-----
<bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
  <property name="interceptors">
    <list>
      <ref bean="maintenanceInterceptor" />
      <ref bean="executeTimeInterceptor" />
    </list>
  </property>
</bean>
-----
</beans>
```

2. User <interceptors> tag in spring configuration file directly:

```
<beans>
-----
<interceptors>
  <interceptor>
    <mapping path="/home" />
    <beans:bean
class="com.durgasoft.interceptors.RequestProcessingTimeInterceptor"></beans:bean>
  </interceptor>
</interceptors>
-----
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. Use <mvc:interceptors> tag in Spring configuration file:

```
<beans>
-----
<mvc:interceptors>
    <bean class="com.javapapers.spring.mvc.interceptor.GreetingInterceptor" />
    <mvc:interceptor>
        <mvc:mapping path="/AnimalList" />
        <bean class="com.durgasoft.interceptor.AnimalInterceptor" />
    </mvc:interceptor>
</mvc:interceptors>
-----
</beans>
```

Example:**index.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="wish"/>
11. </body>
12. </html>
```

helloform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Form </h3>
12. <form method="POST" action="wish">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center"> Durga Software Solutions </h2>
11. <h3 style="color: blue;" align="center"> User Hello Status </h3>
12. <h1 style="color: red;" align="center"> Hello ${uname} </h1>
13. <h1 style="color: blue;" align="center">
14. Start Time: ${startTime}ms<br>
15. End Time : ${endTime}ms<br>
16. HelloController Processing Time : ${totalTime} ms
17. </h1>
18. </body>
19. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

WishController.java

```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.stereotype.Controller;
7. import org.springframework.web.bind.annotation.RequestMapping;
8. import org.springframework.web.bind.annotation.RequestMethod;
9. import org.springframework.web.servlet.ModelAndView;
10.
11.
12. @Controller
13. @RequestMapping("/wish")
14. public class WishController {
15.     @RequestMapping(method=RequestMethod.GET)
16.     public String getForm() {
17.         return "helloform";
18.     }
19.
20.     @RequestMapping(method=RequestMethod.POST)
21.     public ModelAndView wish(HttpServletRequest request, HttpServletResponse response)
22.     {
23.         String uname = request.getParameter("uname");
24.         return new ModelAndView("wish", "uname", uname);
25.     }
}
```

ProcessingTimeInterceptor.java

```
1. package com.durgasoft.interceptor;
2. import javax.servlet.http.HttpServletRequest;
3. import javax.servlet.http.HttpServletResponse;
4.
5. import org.springframework.web.servlet.HandlerInterceptor;
6. import org.springframework.web.servlet.ModelAndView;
7.
8. public class ProcessingTimeInterceptor implements HandlerInterceptor {
9.
10.    @Override
11.    public void afterCompletion(HttpServletRequest arg0, HttpServletResponse arg1, Object arg2, Exception arg3)
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12.      throws Exception {
13.          System.out.println("Response Rendered");
14.      }
15.      @Override
16.      public void postHandle(HttpServletRequest request, HttpServletResponse response, O
    bject handler, ModelAndView modelAndView)
17.          throws Exception {
18.              long startTime = (Long) request.getAttribute("startTime");
19.              request.removeAttribute("startTime");
20.
21.              long endTime = System.currentTimeMillis();
22.              System.out.println("End Time :" + endTime);
23.
24.              System.out.println("Processing Time : " + (endTime - startTime));
25.              modelAndView.addObject("startTime", startTime);
26.              modelAndView.addObject("endTime", endTime);
27.              modelAndView.addObject("totalTime", endTime - startTime);
28.          }
29.      @Override
30.      public boolean preHandle(HttpServletRequest request, HttpServletResponse response,
    Object handler) throws Exception {
31.          long startTime = System.currentTimeMillis();
32.          request.setAttribute("startTime", startTime);
33.          System.out.println("start Time :" + startTime);
34.          return true;
35.      }
36.

```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.    <context:annotation-config/>
12.    <context:component-scan base-package="com.durgasoft.controller"/>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

13. <bean name="processingTimeInterceptor" class="com.durgasoft.interceptor.Processing
   TimeInterceptor"/>
14. <bean class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandle
   rMapping">
15. <property name="interceptors">
16.   <list>
17.     <ref bean="processingTimeInterceptor"/>
18.   </list>
19. </property>
20.</bean>
21. <bean name="viewResolver" class="org.springframework.web.servlet.view.InternalReso
   urceViewResolver">
22.   <property name="prefix" value="/WEB-INF/"/>
23.   <property name="suffix" value=".jsp"/>
24. </bean>
25.</beans>
```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app17</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21.</web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

ViewResolvers In Spring WEB MVC

In Spring MVC, the main intention of view resolvers are to enable us to render models in a browser without tying you to a specific view technology like JSP, Velocity, XML...etc.

To handle views, Spring has provided the following two interfaces

1. ViewResolver
2. View

1. ViewResolver: It will provide mapping between View names and the actual views.

2. View: It will take care of preparing Request and handover to the respective View Technology.

In Spring WEB MVC applications, ViewResolver takes logical name provided by Controller class through DispatcherServlet and return View object point to the actual View page, in this context, DispatcherServlet will access render() method on View object to render the control to view resource.

Spring Web MVC has provided the following ViewResolvers to handle View in Spring WEB MVC applications.

1. AbstractCachingViewResolver
2. XmlViewResolver
3. ResourceBundleViewResolver
4. UrlBasedViewResolver
5. InternalResourceViewResolver
6. VelocityViewResolver/FreeMarkerViewResolver

From the above list of View Resolvers , mainly we are able to use the following ViewResolvers in Spring WEB MVC Applications

1. UrlBasedViewResolver
2. InternalResourceViewResolver
3. XmlViewResolver
4. ResourceBundleViewResolver

1. UrlBasedViewResolver:

This ViewResolver will take view logical name from Controller through DispatcherServlet and searches for the view resources whose name is same as logical view name.

This ViewResolver must required the following three properties configuration.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. viewClass: It will take a particular View Class inorder to render response.

EX: org.springframework.web.servlet.view.JstlView.

2. prefix: It will take the location of the View pages under WEB-INF.

3. suffix: It will take the View page extension.

EX:

```
<beans>
  <bean name="viewResolver"
class="org.springframework.web.servlet.view.UrlBasedViewResolver">
    <property name="viewClass" value="org.springframework.web.servlet.view.JstlView"/>
    <property name="prefix" value="/WEB-INF/pages/" />
    <property name="suffix" value=".jsp"/>
  </bean>
</beans>
```

Note: If we want to use JstlView class as value to the property "viewClass" then we must provide the following jar files in web application "lib" folder.

- 1 .taglibs-standard-impl-1.2.5.jar
2. taglibs-standard-spec-1.2.5.jar

In general, the above jar files are existed in "C:\Tomcat 9.0\webapps\examples\WEB-INF\lib" location.

Example:

index.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC -
   /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="hello"/>
11. </body>
12. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

helloform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: Red;" align="center">Durga Software Solutions</h2>
11. <h3 style="color: Blue;" align="center">User Hello Page</h3>
12. <form method="POST" action="hello">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: Red;" align="center">Durga Software Solutions</h2>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11. <h3 style="color: Blue;" align="center">User Wish Page</h3>
12. <h1 style="color: Red;" align="center">Hello ${uname}</h1>
13. </body>
14. </html>
```



HelloController.java

```

1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.stereotype.Controller;
7. import org.springframework.web.bind.annotation.RequestMapping;
8. import org.springframework.web.bind.annotation.RequestMethod;
9.
10. import org.springframework.web.servlet.ModelAndView;
11.
12.
13. @Controller
14. @RequestMapping("/hello")
15. public class HelloController {
16.     @RequestMapping(method=RequestMethod.GET)
17.     public String showForm() {
18.         return "helloform";
19.     }
20.
21.     @RequestMapping(method=RequestMethod.POST)
22.     public ModelAndView wish(HttpServletRequest request, HttpServletResponse response
23. ) {
24.         String uname = request.getParameter("uname");
25.         return new ModelAndView("wish", "uname", uname);
26.     }
}
```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:p="http://www.springframework.org/schema/p"
5.     xmlns:context="http://www.springframework.org/schema/context"
6.     xsi:schemaLocation=""
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7.    http://www.springframework.org/schema/beans
8.    http://www.springframework.org/schema/beans/spring-beans.xsd
9.    http://www.springframework.org/schema/context
10.   http://www.springframework.org/schema/context/spring-context.xsd">
11.   <context:annotation-config/>
12.   <context:component-scan base-package="com.durgasoft.controller"/>
13.
14.   <bean name="viewResolver" class="org.springframework.web.servlet.view.UrlBasedViewResolver">
15.     <property name="viewClass" value="org.springframework.web.servlet.view.JstlView"/>
16.     <property name="prefix" value="/WEB-INF/pages"/>
17.     <property name="suffix" value=".jsp"/>
18.   </bean>
19. </beans>
```

web.xml

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" id="WebApp_ID" version="4.0">
3.    <display-name>app19</display-name>
4.    <welcome-file-list>
5.      <welcome-file>index.html</welcome-file>
6.      <welcome-file>index.htm</welcome-file>
7.      <welcome-file>index.jsp</welcome-file>
8.      <welcome-file>default.html</welcome-file>
9.      <welcome-file>default.htm</welcome-file>
10.     <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.
13.   <servlet>
14.     <servlet-name>ds</servlet-name>
15.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
16.     <load-on-startup>1</load-on-startup>
17.   </servlet>
18.   <servlet-mapping>
19.     <servlet-name>ds</servlet-name>
20.     <url-pattern>/</url-pattern>
21.   </servlet-mapping>
22. </web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. InternalResourceViewResolver

InternalResourceViewResolver is sub class to UrlBasedViewResolver, it maps jsp and html files which are existed in the WebContent/WEB-INF/ folder.

This ViewResolver will take JstlView as default view, if we want to use Jsp pages as view pages with out Jstl tags then we have to use "InternalResourceView" class as "viewClass".

If we want to use this ViewResolver we must use the following properties.

1.prefix: It will take parent location of the View JSp page, that is, /WEB-INF/

2.suffix: It will take view page extension like .jsp or .html

To configure this ViewResolver in Spring Configuration File we have to use the following XML tags.

EX:

```
<beans>
---
<bean name="viewResolver"
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
    <property name="prefix" value="/WEB-INF/"/>
    <property name="suffix" value=".jsp"/>
</bean>
---
</beans>
```

3. XmlViewResolver:

XmlViewResolver is used to resolve the view names using view beans defined in xml file.

If we want to use this ViewResolver in Spring WEB MVC applications then we have to prepare an XML file with all View Pages configuration by using <bean> tags. In Views XML configuration file we have to configure "JstlView" class as bean with the logical name which is returned by the Controller class, in side JstlView class configuration we have to provide a property "url" with the name and location of the respective View page.

Note: The default name of the Xml file is "view.xml" file.

WEB-INF/views.xml

1. `<?xml version="1.0" encoding="UTF-8"?>`
2. `<beans xmlns="http://www.springframework.org/schema/beans"`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xmlns:p="http://www.springframework.org/schema/p"
5. xmlns:context="http://www.springframework.org/schema/context"
6. xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.   <bean name="helloform" class="org.springframework.web.servlet.view.JstlView">
12.     <property name="url" value="/WEB-INF/pages/helloform.jsp"/>
13.   </bean>
14.   <bean name="wish" class="org.springframework.web.servlet.view.JstlView">
15.     <property name="url" value="/WEB-INF/pages/wish.jsp"/>
16.   </bean>
17. </beans>

```

After providing View configuration XML file then we must configure View Configuration file in Spring Configuration File.

To provide View Configuration File configuration in Spring Configuration file we have to use "location" property with the View configuration XML file location in "XmlViewResolver" configuration.

EX:

ds-servlet.xml

```

<beans>
-----
<bean name="viewResolver" class="org.springframework.web.servlet.view.XmlViewResolver">
  <property name="location" value="/WEB-INF/views.xml"/>
-----
</beans>

```

Example:

index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "
 /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="hello"/>
11. </body>
12. </html>
```

**helloform.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: Red;" align="center">Durga Software Solutions</h2>
11. <h3 style="color: Blue;" align="center">User Hello Page</h3>
12. <form method="POST" action="hello">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

wish.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: Red;" align="center">Durga Software Solutions</h2>
11. <h3 style="color: Blue;" align="center">User Wish Page</h3>
12. <h1 style="color: Red;" align="center">Hello ${uname}</h1>
13. </body>
14. </html>
```

HelloController.java



```
1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.stereotype.Controller;
7. import org.springframework.web.bind.annotation.RequestMapping;
8. import org.springframework.web.bind.annotation.RequestMethod;
9.
10. import org.springframework.web.servlet.ModelAndView;
11.
12.
13. @Controller
14. @RequestMapping("/hello")
15. public class HelloController {
16.     @RequestMapping(method=RequestMethod.GET)
17.     public String showForm() {
18.         return "helloform";
19.     }
20.
21.     @RequestMapping(method=RequestMethod.POST)
22.     public ModelAndView wish(HttpServletRequest request, HttpServletResponse response
23. ) {
23.     String uname = request.getParameter("uname");
24.     return new ModelAndView("wish", "uname", uname);
25.   }
26. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

views.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.   <bean name="helloform" class="org.springframework.web.servlet.view.JstlView">
12.     <property name="url" value="/WEB-INF/pages/helloform.jsp"/>
13.   </bean>
14.   <bean name="wish" class="org.springframework.web.servlet.view.JstlView">
15.     <property name="url" value="/WEB-INF/pages/wish.jsp"/>
16.   </bean>
17. </beans>
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.   <context:annotation-config/>
12.   <context:component-scan base-package="com.durgasoft.controller"/>
13.
14.   <bean name="handlerMapping" class="org.springframework.web.servlet.mvc.annotation
 .DefaultAnnotationHandlerMapping"/>
15.   <bean name="viewResolver" class="org.springframework.web.servlet.view.XmlViewRes
 olver">
16.     <property name="location" value="/WEB-INF/views.xml"/>
17.   </bean>
18. </beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>app19</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.
13.   <servlet>
14.     <servlet-name>ds</servlet-name>
15.     <servlet-class> org.springframework.web.servlet.DispatcherServlet</servlet-class>
16.     <load-on-startup>1</load-on-startup>
17.   </servlet>
18.   <servlet-mapping>
19.     <servlet-name>ds</servlet-name>
20.     <url-pattern>/</url-pattern>
21.   </servlet-mapping>
22. </web-app>
```

4. ResourceBundleViewResolver:

ResourceBundleViewResolver is used to resolve the view names using properties file.

If we want to use this ViewResolver in Spring WEB MVC applications then we have to prepare a properties file under "src" folder with all View Pages configuration by using key-value pairs.

In Views properties file we have to configure "JstlView" class as value with the logical name which is returned by the Controller class in the formate like

logical_Name.(class) = org.springframework.web.servlet.view.JstlView

In Views properties file, we have to provide a key "url" with the name and location of the respective View page with the following format.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

`logical_Name.url = /WEB-INF/pages/file_Name.jsp`**EX:** `src/views.properties``helloform.(class)=org.springframework.web.servlet.view.JstlView
helloform.url=/WEB-INF/pages/helloform.jsp`

After preparing views properties file, we have to configure views properties file in Spring configuration file under "ResourceBundleViewResolver" with the "basename" property.

EX:

```
<beans>  
---  
<bean name="viewResolver"  
      class=" org.springframework.web.servlet.view.ResourceBundleViewResolver">  
    <property name="basename" value="views"/>  
</bean>  
---  
</beans>
```

Example:**index.jsp**

1. `<%@ page language="java" contentType="text/html; charset=ISO-8859-1"`
2. `pageEncoding="ISO-8859-1"%>`
3. `<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">`
4. `<html>`
5. `<head>`
6. `<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">`
7. `<title>Insert title here</title>`
8. `</head>`
9. `<body>`
10. `<jsp:forward page="hello"/>`
11. `</body>`
12. `</html>`

helloform.jsp

1. `<%@ page language="java" contentType="text/html; charset=ISO-8859-1"`
2. `pageEncoding="ISO-8859-1"%>`

CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28 **US NUM:** 4433326786**Mail ID:** durgasoftonlinetraining@gmail.com**WEBSITE:** www.durgasoftonline.com**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
3. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: Red;" align="center">Durga Software Solutions</h2>
11. <h3 style="color: Blue;" align="center">User Hello Page</h3>
12. <form method="POST" action="hello">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SayHello"/></td>
21. </tr>
22. </table>
23. </center>
24. </form>
25. </body>
26. </html>
```

wish.jsp



```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: Red;" align="center">Durga Software Solutions</h2>
11. <h3 style="color: Blue;" align="center">User Wish Page</h3>
12. <h1 style="color: Red;" align="center">>Hello ${uname}</h1>
13. </body>
14. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

HelloController.java

```

1. package com.durgasoft.controller;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.springframework.stereotype.Controller;
7. import org.springframework.web.bind.annotation.RequestMapping;
8. import org.springframework.web.bind.annotation.RequestMethod;
9.
10. import org.springframework.web.servlet.ModelAndView;
11.
12. @Controller
13. @RequestMapping("/hello")
14. public class HelloController {
15.     @RequestMapping(method=RequestMethod.GET)
16.     public String showForm() {
17.
18.         return "helloform";
19.     }
20.     @RequestMapping(method=RequestMethod.POST)
21.     public ModelAndView wish(HttpServletRequest request, HttpServletResponse response
    ) {
22.         String uname = request.getParameter("uname");
23.         return new ModelAndView("wish", "uname", uname);
24.     }
25. }
```

views.properties

- + helloform.(class)=org.springframework.web.servlet.view.JstlView
- + helloform.url=/WEB-INF/pages/helloform.jsp
- + wish.(class)=org.springframework.web.servlet.view.JstlView
- + wish.url=/WEB-INF/pages/wish.jsp

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.     xmlns:p="http://www.springframework.org/schema/p"
5.     xmlns:context="http://www.springframework.org/schema/context"
6.     xsi:schemaLocation="
7.         http://www.springframework.org/schema/beans
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8.      http://www.springframework.org/schema/beans/spring-beans.xsd
9.      http://www.springframework.org/schema/context
10.     http://www.springframework.org/schema/context/spring-context.xsd">
11.     <context:annotation-config/>
12.     <context:component-scan base-package="com.durgasoft.controller"/>
13.     <bean name="viewResolver" class="org.springframework.web.servlet.view.ResourceBu-
    ndleViewResolver">
14.       <property name="basename" value="views"/>
15.     </bean>
16.   </beans>
```

web.xml

```
1.   <?xml version="1.0" encoding="UTF-8"?>
2.   <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
    instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
    org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
    app_4_0.xsd" id="WebApp_ID" version="4.0">
3.     <display-name>app19</display-name>
4.     <welcome-file-list>
5.       <welcome-file>index.html</welcome-file>
6.       <welcome-file>index.htm</welcome-file>
7.       <welcome-file>index.jsp</welcome-file>
8.       <welcome-file>default.html</welcome-file>
9.       <welcome-file>default.htm</welcome-file>
10.      <welcome-file>default.jsp</welcome-file>
11.    </welcome-file-list>
12.
13.    <servlet>
14.      <servlet-name>ds</servlet-name>
15.      <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
16.      <load-on-startup>1</load-on-startup>
17.    </servlet>
18.    <servlet-mapping>
19.      <servlet-name>ds</servlet-name>
20.      <url-pattern>/</url-pattern>
21.    </servlet-mapping>
22.  </web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

File Upload in Spring WEB MVC

IN general, in web applications, it is very frequent to perform upload operations inorder to upload files from client to Server.

To perform File Upload operation in SPring WEB MVC applications, Spring WEB MVC framework has provided some predefined library in the form of "org.springframework.web.multipart.commons.CommonsMultipartResolver"

If we want to perform Upload operation in Spring WEB MVC applications then we have to use the following steps.

1. Prepare a user form with File components:

1. In User form , we must specify "POST" as "method" attribute in form tag.
2. in User form, we must provide "multipart/form-data" as "enctype" attribute in form tag.

EX:

uploadform.jsp

```
<form method="POST" action="upload" enctype="multipart/form-data">
<center>
<table>
<tr>
    <td>File1 </td><td><input type="file" name="file"/></td>
</tr>
</table>
</center>
</form>
```

2. Prepare Controller class with MultipartFile parameter in action method:

EX:UploadController.java:

```
@Controller
public class UploadController {
    ----
    @RequestMapping(value="/upload", method=RequestMethod.POST)
    public ModelAndView uploadFiles(@RequestParam MultipartFile file) {
        String status = "";
        try{
            String file_Name = multipartFile.getOriginalFilename();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

        FileOutputStream fos = new FileOutputStream("E:/uploads/" + file_Name);
        byte[] bt = multipartFile.getBytes();
        fos.write(bt);
        status = "SUCCESS";

    } catch (Exception e) {
        status = "FAILURE";
        e.printStackTrace();
    }
    return new ModelAndView("status", "status", status);
}
}

```

3. Configure CommonsMultipartResolver class as bean with the name "multipartResolver" in spring configuration file:

EX: ds-servlet.xml

```
<bean id="multipartResolver"
      class="org.springframework.web.multipart.commons.CommonsMultipartResolver"/>
```

Note: For File Upload operation, Spring Framework is using Commons File Upload library internally, so that, we must provide the following JAR files in web application lib folder along with all Spring JARs.

commons-fileupload-1.4.jar
commons-io-2.6.jar

Example:

index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="upload"/>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

11. </body>
12. </html>

uploadform.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
2. pageEncoding="ISO-8859-1"%>  
3. <!DOCTYPE html>  
4. <html>  
5. <head>  
6. <meta charset="ISO-8859-1">  
7. <title>Insert title here</title>  
8. </head>  
9. <body>  
10. <h2 style="color: red;" align="center">File Upload Form</h2>  
11. <form method="POST" action="upload" enctype="multipart/form-data">  
12. <center>  
13. <table>  
14. <tr>  
15. <td>File1</td><td><input type="file" name="file"/></td>  
16. </tr>  
17. <tr>  
18. <td>File2</td><td><input type="file" name="file"/></td>  
19. </tr>  
20. <tr>  
21. <td>File3</td><td><input type="file" name="file"/></td>  
22. </tr>  
23. <tr>  
24. <td><input type="submit" value="Upload"/></td>  
25. </tr>  
26. </table>  
27. </center>  
28. </form>  
29. </body>  
30. </html>
```

status.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
2. pageEncoding="ISO-8859-1"%>  
3. <!DOCTYPE html>  
4. <html>  
5. <head>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h1 style="color: red;" align="center">Files Upload Status: ${status}</h1>
11. </body>
12. </html>
```

**UploadController.java**

```
1. package com.durgasoft.controller;
2.
3. import java.io.FileOutputStream;
4.
5. import org.springframework.stereotype.Controller;
6. import org.springframework.web.bind.annotation.RequestMapping;
7. import org.springframework.web.bind.annotation.RequestMethod;
8. import org.springframework.web.bind.annotation.RequestParam;
9. import org.springframework.web.multipart.MultipartFile;
10. import org.springframework.web.servlet.ModelAndView;
11.
12. @Controller
13. public class UploadController {
14.     @RequestMapping(value="/upload", method=RequestMethod.GET)
15.     public String showUploadForm() {
16.         return "uploadform";
17.     }
18.
19.     @RequestMapping(value="/upload", method=RequestMethod.POST)
20.     public ModelAndView uploadFiles(@RequestParam MultipartFile[] file) {
21.         String status = "";
22.
23.         try {
24.             for(MultipartFile multipartFile: file) {
25.                 String file_Name = multipartFile.getOriginalFilename();
26.                 FileOutputStream fos = new FileOutputStream("E:/uploads/"+file_Name);
27.                 byte[] bt = multipartFile.getBytes();
28.                 fos.write(bt);
29.                 status = "SUCCESS";
30.             }
31.
32.         } catch (Exception e) {
33.             status = "FAILURE";
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

34.         e.printStackTrace();
35.     }
36.     return new ModelAndView("status", "status", status);
37. }
38.

```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <context:component-scan base-package = "com.durgasoft.controller" />
13.
14. <bean id="multipartResolver"
15.       class="org.springframework.web.multipart.commons.CommonsMultipartResolver"/>
16.
17. <bean class = "org.springframework.web.servlet.view.InternalResourceViewResolver">
18.   <property name = "prefix" value = "/WEB-INF//" />
19.   <property name = "suffix" value = ".jsp" />
20. </bean>
21.
22.</beans>

```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3. <display-name>exception_handling_web</display-name>
4. <welcome-file-list>
5.   <welcome-file>index.html</welcome-file>
6.   <welcome-file>index.htm</welcome-file>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7. <welcome-file>index.jsp</welcome-file>
8. <welcome-file>default.html</welcome-file>
9. <welcome-file>default.htm</welcome-file>
10. <welcome-file>default.jsp</welcome-file>
11. </welcome-file-list>
12. <servlet>
13.   <servlet-name>ds</servlet-name>
14.   <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.   <load-on-startup>1</load-on-startup>
16. </servlet>
17.
18. <servlet-mapping>
19.   <servlet-name>ds</servlet-name>
20.   <url-pattern>/</url-pattern>
21. </servlet-mapping>
22.</web-app>
```

DURGAS

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

File Downloading in Spring WEB MVC

In general, in web applications, it is very frequent operation to perform download operation.

If we want to perform Download operation in Spring WEB MVC Application then we have to use the following steps.

1. Prepare a File with "Download" button and with the downloadable resource.

EX:

downloadpage.jsp

```
<form method="post" action="download">

<br>
<input type="submit" value="Download"/></td>
</form>
```

2. Prepare Controller class:

IN Controller class and in action method we have to use the following steps.

- a) Set Content Type as "APPLICATION/OCTET-STREAM".
`response.setContentType("APPLICATION/OCTET-STREAM");`
- b) Set "content-disposition" response header to response object.
`response.setHeader("Content-Disposition", "attachment;filename=\""+fileName+"\"");`
- c) Create File and FileInputStream object with the downloadable resource.
`File file = new File("E:\\images\\Java_Python.jpg");
FileInputStream fis = new FileInputStream(file);`
- d) Create OutputStream from response object.
`OutputStream os = response.getOutputStream()`
- e) Get bit by bit from FileInputStream and write bit by bit to OutputStream
`int val = fis.read();
while(val != -1) {
os.write(val);
val = fis.read();
}`

Example:

index.jsp

1. `<%@ page language="java" contentType="text/html; charset=ISO-8859-1"`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.  pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="download"/>
11. </body>
12. </html>
```

downloadpage.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.  pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html>
4. <html>
5. <head>
6. <meta charset="ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <form method="post" action="download">
11. <center>
12. <table>
13. <tr>
14.  <td></td>
15. </tr>
16. <tr>
17.  <td><input type="submit" value="Download"/></td>
18. </tr>
19. </table>
20. </center>
21. </form>
22. </body>
23. </html>
```

DownloadController.java

```
1. package com.durgasoft.controller;
2.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. import java.io.File;
4. import java.io.FileInputStream;
5. import java.io.OutputStream;
6.
7. import javax.servlet.http.HttpServletRequest;
8. import javax.servlet.http.HttpServletResponse;
9.
10. import org.springframework.stereotype.Controller;
11. import org.springframework.web.bind.annotation.RequestMapping;
12. import org.springframework.web.bind.annotation.RequestMethod;
13. import org.springframework.web.servlet.ModelAndView;
14.
15. @Controller
16. public class DownloadController {
17.     @RequestMapping(value="/download",method=RequestMethod.GET)
18.     public String showDownloadPage() {
19.         return "downloadpage";
20.     }
21.     @RequestMapping(value="/download", method=RequestMethod.POST)
22.     public void download(HttpServletRequest request, HttpServletResponse response) throws
Exception {
23.         response.setContentType("APPLICATION/OCTET-STREAM");
24.         File file = new File("E:\\images\\Java_Python.jpg");
25.         FileInputStream fis = new FileInputStream(file);
26.         String fileName = file.getName();
27.         response.setHeader("Content-Disposition", "attachment;filename=\""+fileName+"\"");
28.
29.         OutputStream os = response.getOutputStream();
30.         int val = fis.read();
31.
32.         while(val != -1) {
33.             os.write(val);
34.             val = fis.read();
35.         }
36.         fis.close();
37.         os.close();
38.     }
39. }
```

ds-servlet.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4. xmlns:p="http://www.springframework.org/schema/p"
5. xmlns:context="http://www.springframework.org/schema/context"
6. xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <context:component-scan base-package = "com.durgasoft.controller" />
13. <bean class = "org.springframework.web.servlet.view.InternalResourceViewResolver">
14.   <property name = "prefix" value = "/WEB-INF//" />
15.   <property name = "suffix" value = ".jsp" />
16. </bean>
17.
18.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>exception_handling_web</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.   <servlet-mapping>
18.     <servlet-name>ds</servlet-name>
19.     <url-pattern>/</url-pattern>
20.   </servlet-mapping>
21.</web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Internationalization[I18N]

Internationalization is the process of designing Java application on the basis of Locality is called as Internationalization.

Java is providing very good Internationalization support in Java applications due to UNICODE representations and the predefined library like `java.util.Locale` and some of the predefined classes from `java.text` package.

To provide Internationalization in Spring WEB MVC applications , Spring has provided very good environment with the following two beans.

1. `SessionLocaleResolver`
2. `LocaleChangeInterceptor`

1. SessionLocaleResolver:

`SessionLocaleResolver` resolves locales by inspecting a predefined attribute in a user's session. If the session attribute doesn't exist, this locale resolver determines the default locale from the `accept-language` HTTP header.

```
<bean id="localeResolver"
class="org.springframework.web.servlet.i18n.SessionLocaleResolver">
    <property name="defaultLocale" value="en"/>
</bean>
```

2. LocaleChangeInterceptor:

`LocaleChangeInterceptor` interceptor detects if a special parameter is present in the current HTTP request. The parameter name can be customized with the `paramName` property of this interceptor. If such a parameter is present in the current request, this interceptor changes the user's locale according to the parameter value.

```
<bean id="localeChangeInterceptor"
class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">
    <property name="paramName" value="lang" />
</bean>

<bean
class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping">
    <property name="interceptors">
        <list>
            <ref bean="localeChangeInterceptor" />
        </list>
    </property>
</bean>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

To implement Internationalization in Spring WEB MVC application then we have to use the following steps.

1. Prepare properties file for each and every local to which we want to support.

In Internationalization , we have to prepare a seperate properties file with the locale respective messages in the form of Key-Value pairs, where keys must be in English and the values must be in the respective language letters in the form of UNOCODE.

In general, the name format of the properties file is "baseName_lang.properties".

EX:

messages_en.properties

uname = User Name
upwd = User Password
messages_it.properties

uname = Usere Nameo
upwd = Usere Passwordo

2. Prepare User forms by getting messages from the properties file:

To get messages from properties files we have to use the following tag from Spring tag library with the URI "http://www.springframework.org/tags".

<spring:message code="--"/>

Where code attribute will take key of the message defined in properties file.

registrationform.jsp

```
1. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
2. <%@taglib uri="http://www.springframework.org/tags" prefix="spring"%>
3. <html>
4. <body>
5. <form:form method="POST" action="reg" commandName="user">
6. <center>
7. <table>
8. <tr>
9.   <td><spring:message code="uname"/></td>
10.  <td><form:input path="uname"/></td>
11. </tr>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12.<tr>
13. <td><spring:message code="upwd"/></td>
14. <td><form:password path="upwd"/></td>
15.</tr>
16.<tr>
17. <td><spring:message code="uemail"/></td>
18. <td><form:input path="uemail"/></td>
19.</tr>
20.<tr>
21. <td><spring:message code="umobile"/></td>
22. <td><form:input path="umobile"/></td>
23.</tr>
24.<tr>
25. <td><input type="submit" value="Registration"/></td>
26.</tr>
27.</table>
28.</center>
29.</form:form>
30.</body>
31.</html>

```

3. Configure "SessionLocaleResolver", "LocaleChangeInterceptor" and " ResourceBundleMessageSource" in Spring Configuration File:

Where the purpose of SessionLocaleResolver is to resolves locales by inspecting a predefined attribute in a user's session. If the session attribute doesn't exist, this locale resolver determines the default locale from the accept-language HTTP header.

Where the purpose of "LocaleChangeInterceptor" is to detects if a special parameter is present in the current HTTP request. The parameter name can be customized with the paramName property of this interceptor. If such a parameter is present in the current request, this interceptor changes the user's locale according to the parameter value.

Where the purpose of "ResourceBundleMessageSource" is to resolve the properties file on the basis of the provided "basename" property inorder to submit values to the User forms.

ds-servlet.xml

```

1. <beans>
2.   <context:component-scan base-package="com.durgasoft" />
3.   <bean class="org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter" />
4.

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5.   <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
6.     <property name="prefix" value="/WEB-INF/views/" />
7.     <property name="suffix" value=".jsp" />
8.   </bean>
9.
10.  <bean id="messageSource" class="org.springframework.context.support.ResourceBundl
eMessageSource">
11.    <property name="basename" value="messages" />
12.  </bean>
13.
14.  <bean id="localeResolver" class="org.springframework.web.servlet.i18n.SessionLocale
Resolver">
15.    <property name="defaultLocale" value="en" />
16.  </bean>
17.
18.  <bean id="localeChangeInterceptor" class="org.springframework.web.servlet.i18n.Local
eChangeInterceptor">
19.    <property name="paramName" value="lang" />
20.  </bean>
21.  <bean class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandle
rMapping">
22.    <property name="interceptors">
23.      <list>
24.        <ref bean="localeChangeInterceptor" />
25.      </list>
26.    </property>
27.  </bean>
28.
29.</beans>
```

Example:**messages_en.properties**

- ⊕ title1 = Durga Software Solutions
- ⊕ title2 = User Registration Form
- ⊕ uname = User Name
- ⊕ upwd = User Password
- ⊕ uemail = User Email Id
- ⊕ umobile = User Mobile No

messages_it.properties

- ⊕ title1 = Durga Software Solutions
- ⊕ title2 = Usere Registratione Formo
- ⊕ uname = Usero Nameo
- ⊕ upwd = Passwordo

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28


US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- uemail = Usere Emailo
- umobile = Usero Mobileo

messages_hi.properties

- title1 = Durga Software Solutions
- title2 = User Registration Form
- uname = Upayuktha Naam
- upwd = Upayuktha Rahsya Naam
- uemail = Upayuktha Vidyuth Naam
- umobile = Upayuktha Charavani Sakya

index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <center>
11. <h2 style="color: red;" align="center">User Registration Form</h2>
12. <h3>
13. <a href="/i18napp/reg?lang=en">English</a>|<a href="/i18napp/reg?lang=it">Italian</a>|<a href="/i18napp/reg?lang=hi">Hindi</a>
14. </h3>
15. </center>
16. </body>
17. </html>

```

registrationform.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
4. <%@taglib uri="http://www.springframework.org/tags" prefix="spring"%>
5. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
6. <html>
7. <head>
8. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
9. <title>Insert title here</title>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10.</head>
11.<body>
12.<h2 style="color: red;" align="center">
13.<spring:message code="title1"/>
14.</h2>
15.<h3 style="color: blue;" align="center">
16.<spring:message code="title2"/>
17.</h3>
18.<form:form method="POST" action="reg" commandName="user">
19.<center>
20.<table>
21.<tr>
22. <td><spring:message code="uname"/></td>
23. <td><form:input path="uname"/></td>
24.</tr>
25.<tr>
26. <td><spring:message code="upwd"/></td>
27. <td><form:password path="upwd"/></td>
28.</tr>
29.<tr>
30. <td><spring:message code="uemail"/></td>
31. <td><form:input path="uemail"/></td>
32.</tr>
33.<tr>
34. <td><spring:message code="umobile"/></td>
35. <td><form:input path="umobile"/></td>
36.</tr>
37.<tr>
38. <td><input type="submit" value="Registration"/></td>
39.</tr>
40.</table>
41.</center>
42.</form:form>
43.</body>
44.</html>
```

registrationdetails.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <%@taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
4. <%@taglib uri="http://www.springframework.org/tags" prefix="spring"%>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. <!DOCTYPE html PUBLIC "-
  /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
6. <html>
7. <head>
8. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
9. <title>Insert title here</title>
10. </head>
11. <body>
12. <h2 style="color: red;" align="center">
13. <spring:message code="title1"/>
14. </h2>
15. <h3 style="color: blue;" align="center">
16. <spring:message code="title2"/>
17. </h3>
18. <center>
19. <table border="1">
20. <tr>
21.   <td><spring:message code="uname"/></td>
22.   <td>${user.uname}</td>
23. </tr>
24. <tr>
25.   <td><spring:message code="upwd"/></td>
26.   <td>${user.upwd}</td>
27. </tr>
28. <tr>
29.   <td><spring:message code="uemail"/></td>
30.   <td>${user.uemail}</td>
31. </tr>
32. <tr>
33.   <td><spring:message code="umobile"/></td>
34.   <td>${user.umobile}</td>
35. </tr>
36. </table>
37. </center>
38. </body>
39. </html>
```

I18NController.java

```
1. package com.durgasoft.controller;
2.
3. import org.springframework.stereotype.Controller;
4. import org.springframework.web.bind.annotation.RequestMapping;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. import org.springframework.web.bind.annotation.RequestMethod;
6. import org.springframework.web.servlet.ModelAndView;
7. import com.durgasoft.command.User;
8.
9. @Controller
10. public class I18nController {
11.     @RequestMapping(value="/reg", method=RequestMethod.GET)
12.     public ModelAndView showForm() {
13.         return new ModelAndView("registrationform", "user", new User());
14.     }
15.
16.     @RequestMapping(value="/reg", method=RequestMethod.POST)
17.     public ModelAndView registration(User user) {
18.         return new ModelAndView("registrationdetails", "user", user);
19.     }
20. }
```

User.java



```
1. package com.durgasoft.command;
2.
3. public class User {
4.     private String uname;
5.     private String upwd;
6.     private String uemail;
7.     private String umobile;
8.
9.     public String getUname() {
10.         return uname;
11.     }
12.     public void setUname(String uname) {
13.         this.uname = uname;
14.     }
15.     public String getUpwd() {
16.         return upwd;
17.     }
18.     public void setUpwd(String upwd) {
19.         this.upwd = upwd;
20.     }
21.     public String getUemail() {
22.         return uemail;
23.     }
24.     public void setUemail(String uemail) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
25.     this.uemail = uemail;
26. }
27. public String getUmobile() {
28.     return umobile;
29. }
30. public void setUmobile(String umobile) {
31.     this.umobile = umobile;
32. }
33.}
```

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xsi:schemaLocation="
7.     http://www.springframework.org/schema/beans
8.     http://www.springframework.org/schema/beans/spring-beans.xsd
9.     http://www.springframework.org/schema/context
10.    http://www.springframework.org/schema/context/spring-context.xsd">
11.
12. <context:component-scan base-package = "com.durgasoft.controller" />
13.
14. <bean id="messageSource" class="org.springframework.context.support.ResourceBun
  dleMessageSource">
15.   <property name="basename" value="/com/durgasoft/resources/messages" />
16. </bean>
17.
18. <bean id="localeResolver" class="org.springframework.web.servlet.i18n.SessionLocale
  Resolver">
19.   <property name="defaultLocale" value="en" />
20. </bean>
21.
22. <bean id="localeChangelInterceptor" class="org.springframework.web.servlet.i18n.Local
  eChangelInterceptor">
23.   <property name="paramName" value="lang" />
24. </bean>
25.
26. <bean class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandle
  rMapping">
27.   <property name="interceptors">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
28.      <list>
29.          <ref bean="localeChangeInterceptor" />
30.      </list>
31.  </property>
32. </bean>
33.
34. <bean class = "org.springframework.web.servlet.view.InternalResourceViewResolver">
35.   <property name = "prefix" value = "/WEB-INF/" />
36.   <property name = "suffix" value = ".jsp" />
37. </bean>
38.
39.</beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>i18napp</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <servlet-name>ds</servlet-name>
14.     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
15.     <load-on-startup>1</load-on-startup>
16.   </servlet>
17.
18.   <servlet-mapping>
19.     <servlet-name>ds</servlet-name>
20.     <url-pattern>/</url-pattern>
21.   </servlet-mapping>
22.</web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

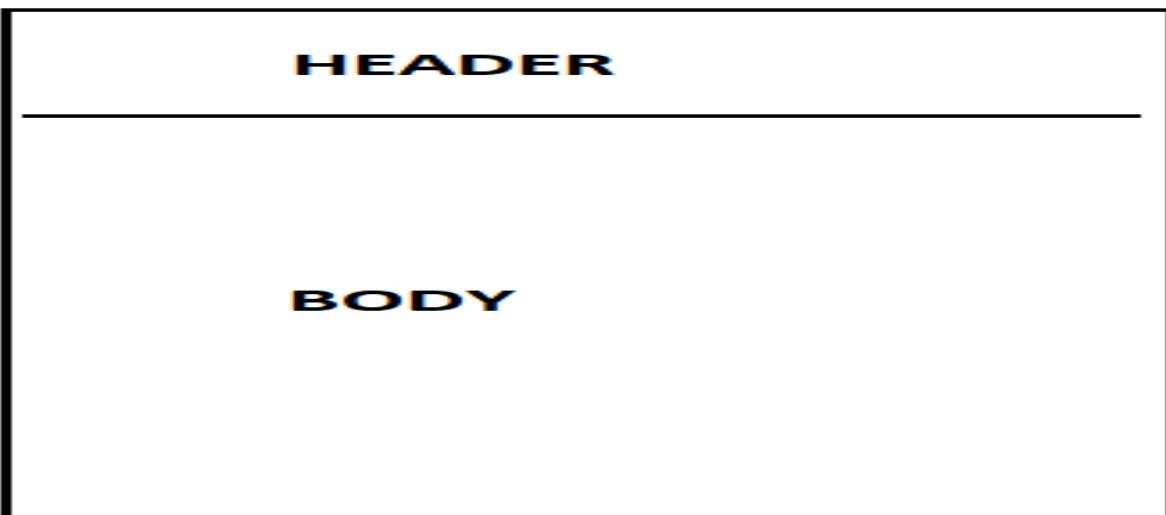
SPRING MVC with TILES

Spring WEB MVC with Tiles Integration:

In general, in web applications, we have to organize the web pages in a standard mode inorder to improve Look and Feel.

To manage all the web pages in standard manner we have to use templates.

In general, we will use the following templates to prepare web pages in web applications.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

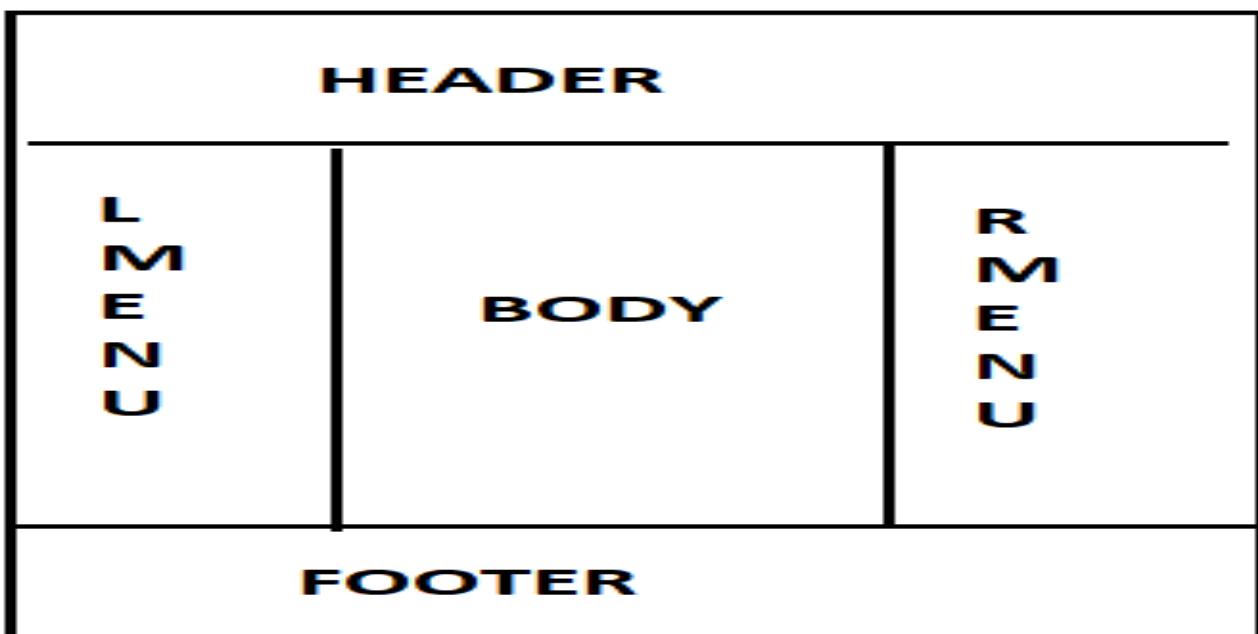
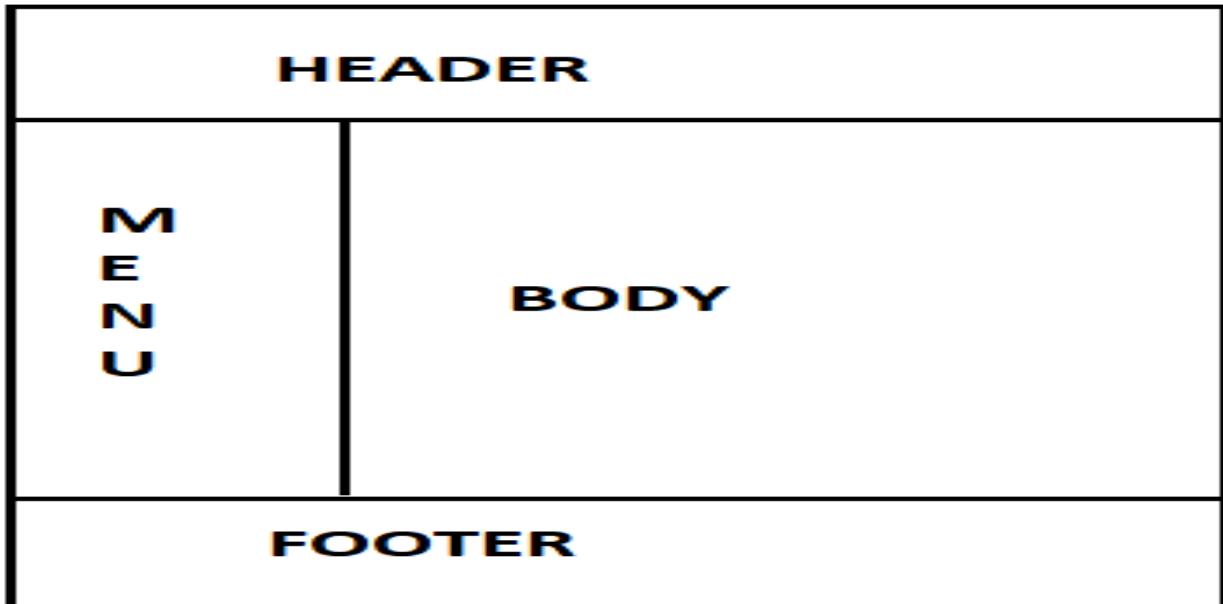
Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU



To manage all the web pages in the above provided templates and to manage flow of execution between all the web pages we have to use a Apache provided Tiles framework.

Apache Software Foundations has provided the complete Tiles Framework in the form of the following JAR files.

- ✓ tiles-api-2.2.2.jar

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ tiles-core-2.2.2.jar
- ✓ tiles-jsp-2.2.2.jar
- ✓ tiles-servlet-2.2.2.jar
- ✓ tiles-template-2.2.2.jar

If we want to use Tiles Framework in our web applications then we have to provide the above Jar files in web application lib folder.

The web frameworks like Struts, JSF, Xwork2 are using already Tiles Framework to prepare web applications.

Spring web MVC framework is also providing in built support for Tiles Framework Integration inorder to prepare web applications.

If we want to use Tiles Framework in Spring web MVC applications then we have to use the following steps.

1. Prepare Template page by using Tiles tags.
2. Prepare Tiles pages.
3. Prepare Tiles definitions.
4. Prepare Controller class.

1. Prepare Template page by using Tiles tags:

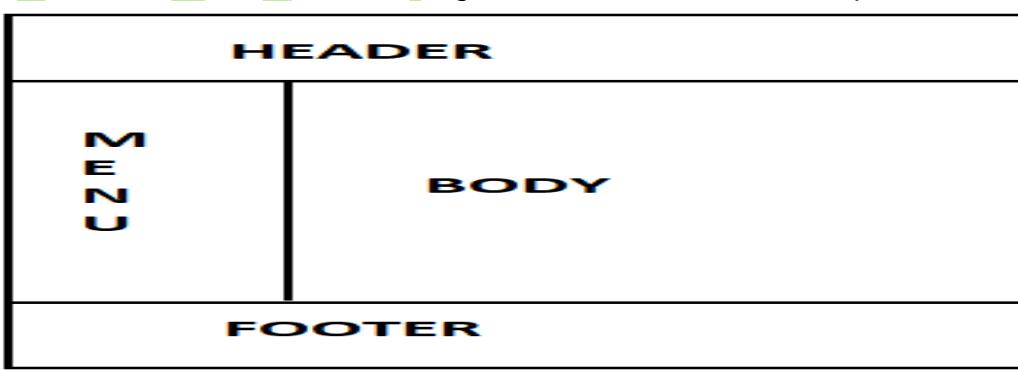
The main intention of Template page is to define a standard template for web pages.

To prepare Template pages we have to use the following Tiles tag library which is available with the URI "http://tiles.apache.org/tags-tiles".

```
<tiles:insertAttribute name="--"/>
```

This tag can be used to define tiles logical names in Template.

Where "name" attribute will take logical name of the tile in Template.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:

```
1. <%@taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>
2. <html>
3. <body>
4. <table width="100%" height="100%">
5. <tr height="20%">
6.   <td colspan="2" align="center">
7.     <tiles:insertAttribute name="header"/>
8.   </td>
9. </tr>
10. <tr height="60%">
11.   <td width="20%">
12.     <tiles:insertAttribute name="menu"/>
13.   </td>
14.   <td width="80%">
15.     <tiles:insertAttribute name="body"/>
16.   </td>
17. </tr>
18. <tr height="15%">
19.   <td colspan="2" align="center">
20.     <tiles:insertAttribute name="footer"/>
21.   </td>
22. </tr>
23. </table>
24. </body>
25. </html>
```

2. Prepare Tiles pages:

In Tiles based web applications, Tile is a JSP page it able to represent Header, Footer, Menu, Body,.....

EX:**header.jsp**

```
1. <html>
2. <body>
3. <h1 style="color: white;">DURGA SOFTWARE SOLUTIONS</h1>
4. </body>
5. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

menu.jsp

```
1. <html>
2. <body>
3. <br>
4. <h3>
5. <a href="add">Add Student</a><br><br>
6. <a href="search">Search Student</a><br><br>
7. <a href="delete">Delete Student</a>
8. </h3>
9. </body>
10.</html>
```

3. Prepare Tiles definitions

The main intention of Tiles Definitions is to define the combination of tiles pages in the form of Definitions.

In Tiles based web applications, we have to define all tiles definitions in an xml file by using the following xml tags.

```
<!DOCTYPE ----- >
<tiles-definitions>
<definition name="--" template="--">
  <put-attribute name="--" value="--"/>
  -----
</definition>
-----
</tiles-definitions>
```

- ✓ Where `<tiles-definitions>` is root tags, it will include no of definitions.
- ✓ Where `<definition>` tag is able to provide single definition which includes tiles pages.
- ✓ Where "name" attribute in `<definition>` tag will take definition name.
- ✓ Where "template" attribute in `<definition>` tag will take the name and location of the layout page
- ✓ Where `<put-attribute>` tag will assign tile JSP page to the respective logical name of the tile in JSP page.
- ✓ Where "name" in `<put-attribute>` will take logical name of the tile.
- ✓ Where "value" attribute will take the name and location of the tile JSP page.

In Tiles definitions file, we are able to extend one definition to another definition by using "extends" attribute inorder to reuse tiles configurations and we are able to override one tile configuration to another tiles configuration just like normal inheritance and method overriding.

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

Note: We are able to get DOCTYPE definition in tiles definitions file by using the following dtd file from tiles-core-2.2.2.jar.

org\apache\tiles\resources\tiles-config_2_1.dtd

EX:

tiles-defs.xml



```

1. <?xml version="1.0" encoding="UTF-8"?>
2.
3. <!DOCTYPE tiles-definitions PUBLIC "-
   //Apache Software Foundation//DTD Tiles Configuration 2.1//EN"
4. "http://tiles.apache.org/dtds/tiles-config_2_1.dtd">
5.
6. <tiles-definitions>
7.   <definition name="welcomeDef" template="/WEB-INF/layout.jsp">
8.     <put-attribute name="header" value="/WEB-INF/header.jsp"/>
9.     <put-attribute name="menu" value="/WEB-INF/menu.jsp"/>
10.    <put-attribute name="body" value="/WEB-INF/welcome.jsp"/>
11.    <put-attribute name="footer" value="/WEB-INF/footer.jsp"/>
12.  </definition>
13.  <definition name="addDef" extends="welcomeDef">
14.    <put-attribute name="body" value="/WEB-INF/addstudent.jsp"/>
15.  </definition>
16.  <definition name="searchDef" extends="welcomeDef">
17.    <put-attribute name="body" value="/WEB-INF/searchstudent.jsp"/>
18.  </definition>
19.  <definition name="deleteDef" extends="welcomeDef">
20.    <put-attribute name="body" value="/WEB-INF/deletestudent.jsp"/>
21.  </definition>
22.  <definition name="statusDef" extends="welcomeDef">
23.    <put-attribute name="body" value="/WEB-INF/status.jsp"/>
24.  </definition>
25. </tiles-definitions>
```

4. Prepare Controller class.

In Tiles based applications, we will prepare Controller classes as like normal Controller classes , but, the respective Business methods must return tiles definitions logical name only instead of jsp pages names.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:**StudentController.java**

```
1. @Controller
2. public class StudentController {
3.
4.     @RequestMapping(value="/welcome", method=RequestMethod.GET)
5.     public String welcome() {
6.         return "welcomeDef";
7.     }
8.
9.     @RequestMapping(value="/add", method=RequestMethod.GET)
10.    public ModelAndView addStudent() {
11.        return new ModelAndView("addDef", "student", new Student());
12.    }
13.
14.    @RequestMapping(value="/search", method=RequestMethod.GET)
15.    public ModelAndView searchStudent() {
16.        return new ModelAndView("searchDef", "student", new Student());
17.    }
18.
19.    @RequestMapping(value="/delete", method=RequestMethod.GET)
20.    public ModelAndView deleteStudent() {
21.        return new ModelAndView("deleteDef", "student", new Student());
22.    }
23.
24.    @RequestMapping(value="/add", method=RequestMethod.POST)
25.    public ModelAndView add(Student student) {
26.        String status = studentService.addStudent(student);
27.        return new ModelAndView("statusDef", "status", status);
28.    }
29.
30.    @RequestMapping(value="/search", method=RequestMethod.POST)
31.    public ModelAndView search(Student student) {
32.        Student std = studentService.searchStudent(student.getSid());
33.        if(std == null) {
34.            return new ModelAndView("statusDef", "status", "Student Not Existed");
35.        }else {
36.            return new ModelAndView("studentDetailsDef", "student", std);
37.        }
38.    }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

39. @RequestMapping(value="/delete", method=RequestMethod.POST)
40. public ModelAndView delete(Student student) {
41.     String status = studentService.deleteStudent(student.getSid());
42.     return new ModelAndView("statusDef", "status", status);
43. }
44.

```

Prepare Spring Configuration File:

In Spring configuration file we have to provide the following configuration details.

1. UrlBasedViewResolver with TilesView class.
2. TilesConfigurer with "definitions" property of list type with tiles definitions xml file.

EX:

ds-servlet.xml

```

1. <beans>
2. -----
3.     <bean id="viewResolver"
4.         class="org.springframework.web.servlet.view.UrlBasedViewResolver">
5.         <property name="viewClass">
6.             <value>
7.                 org.springframework.web.servlet.view.tiles2.TilesView
8.             </value>
9.         </property>
10.    </bean>
11.
12.    <bean id="tilesConfigurer"
13.        class="org.springframework.web.servlet.view.tiles2.TilesConfigurer">
14.        <property name="definitions">
15.            <list>
16.                <value>/WEB-INF/tiles-defs.xml</value>
17.            </list>
18.        </property>
19.    </bean>
20. -----
21. </beans>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**index.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <jsp:forward page="welcome"/>
11. </body>
12. </html>
```

layout.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3.
4. <%@taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>
5. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
6. <html>
7. <head>
8. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
9. <title>Insert title here</title>
10. </head>
11. <body>
12. <table width="100%" height="550">
13. <tr height="20%">
14.   <td colspan="2" align="center" bgcolor="maroon">
15.     <tiles:insertAttribute name="header"/>
16.   </td>
17. </tr>
18. <tr height="60%">
19.   <td width="20%" bgcolor="lightyellow">
20.     <tiles:insertAttribute name="menu"/>
21.   </td>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
22. <td width="80%" bgcolor="lightblue">
23.   <tiles:insertAttribute name="body"/>
24. </td>
25. </tr>
26. <tr height="15%">
27.   <td colspan="2" align="center" bgcolor="blue">
28.     <tiles:insertAttribute name="footer"/>
29.   </td>
30. </tr>
31. </table>
32. </body>
33. </html>
```

header.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h1 style="color: white;">DURGA SOFTWARE SOLUTIONS</h1>
11. </body>
12. </html>
```

menu.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br>
11. <h3>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
12.<a href="add">Add Student</a><br><br>
13.<a href="search">Search Student</a><br><br>
14.<a href="delete">Delete Student</a>
15.</h3>
16.</body>
17.</html>
```

welcome.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10.<br><br><br>
11.<h1 style="color: red;">
12.<marquee>
13.Welcome To Durga Software Solutions
14.</marquee>
15.</h1>
16.</body>
17.</html>
```

footer.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10.<h3 style="color: white" align="center">
11.Durgasoft India Pvt Ltd., 202, HMDA, Mitrivanam, Ameerpet, Hyd-38
12.</h3>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

13. </body>
14. </html>

addstudent.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3.
4. <%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
5. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
6. <html>
7. <head>
8. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
9. <title>Insert title here</title>
10. </head>
11. <body><br><br>
12. <form:form method="POST" action="add" commandName="student">
13. <center>
14. <table>
15. <tr>
16.   <td>Student Id</td>
17.   <td><form:input path="sid"/></td>
18. </tr>
19. <tr>
20.   <td>Student Name</td>
21.   <td><form:input path="sname"/></td>
22. </tr>
23. <tr>
24.   <td>Student Address</td>
25.   <td><form:input path="saddr"/></td>
26. </tr>
27. <tr>
28.   <td><input type="submit" value="ADD"/></td>
29. </tr>
30. </table>
31. </center>
32. </form:form>
33. </body>
34. </html>
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

searchstudent.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3.
4. <%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
5. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
6. <html>
7. <head>
8. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
9. <title>Insert title here</title>
10. </head>
11. <body><br><br>
12. <form:form method="POST" action="search" commandName="student">
13. <center>
14. <table>
15. <tr>
16.   <td>Student Id</td>
17.   <td><form:input path="sid"/></td>
18. </tr>
19. <tr>
20.   <td><input type="submit" value="SEARCH"/></td>
21. </tr>
22. </table>
23. </center>
24. </form:form>
25. </body>
26. </html>
```

deletestudent.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
4. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <body><br><br>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11. <form:form method="POST" action="delete" commandName="student">
12. <center>
13. <table>
14. <tr>
15.   <td>Student Id</td>
16.   <td><form:input path="sid"/></td>
17. </tr>
18. <tr>
19.   <td><input type="submit" value="DELETE"/></td>
20. </tr>
21. </table>
22. </center>
23. </form:form>
24. </body>
25. </html>
```

studentdetails.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br>
11. <center>
12. <table border="1">
13. <tr>
14.   <td>Student Id</td>
15.   <td>${student.sid}</td>
16. </tr>
17. <tr>
18.   <td>Student Name</td>
19.   <td>${student.sname}</td>
20. </tr>
21. <tr>
22.   <td>Student Address</td>
23.   <td>${student.saddr}</td>
24. </tr>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

25. </table>
 26. </center>
 27. </body>
 28. </html>

status.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//
 /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br>
11. <h1 style="color: red;" align="center">${status}</h1>
12. </body>
13. </html>
```

StudentController.java

```
1. package com.durgasoft.controller;
2.
3. import org.springframework.beans.factory.annotation.Autowired;
4. import org.springframework.stereotype.Controller;
5. import org.springframework.web.bind.annotation.RequestMapping;
6. import org.springframework.web.bind.annotation.RequestMethod;
7. import org.springframework.web.servlet.ModelAndView;
8.
9. import com.durgasoft.beans.Student;
10. import com.durgasoft.service.StudentService;
11.
12. @Controller
13. public class StudentController {
14.
15.   @Autowired
16.   private StudentService studentService;
17.
18.   @RequestMapping(value="/welcome", method=RequestMethod.GET)
19.   public String welcome() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
20.     return "welcomeDef";
21. }
22.
23. @RequestMapping(value="/add", method=RequestMethod.GET)
24. public ModelAndView addStudent() {
25.     return new ModelAndView("addDef", "student", new Student());
26. }
27.
28. @RequestMapping(value="/search", method=RequestMethod.GET)
29. public ModelAndView searchStudent() {
30.     return new ModelAndView("searchDef", "student", new Student());
31. }
32.
33. @RequestMapping(value="/delete", method=RequestMethod.GET)
34. public ModelAndView deleteStudent() {
35.     return new ModelAndView("deleteDef", "student", new Student());
36. }
37.
38. @RequestMapping(value="/add", method=RequestMethod.POST)
39. public ModelAndView add(Student student) {
40.     String status = studentService.addStudent(student);
41.     return new ModelAndView("statusDef", "status", status);
42. }
43.
44. @RequestMapping(value="/search", method=RequestMethod.POST)
45. public ModelAndView search(Student student) {
46.     Student std = studentService.searchStudent(student.getId());
47.     if(std == null) {
48.         return new ModelAndView("statusDef", "status", "Student Not Existed");
49.     }else {
50.         return new ModelAndView("studentDetailsDef", "student", std);
51.     }
52. }
53. @RequestMapping(value="/delete", method=RequestMethod.POST)
54. public ModelAndView delete(Student student) {
55.     String status = studentService.deleteStudent(student.getId());
56.     return new ModelAndView("statusDef", "status", status);
57. }
58. }
```

StudentService.java

```
1. package com.durgasoft.service;
```

CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28**US NUM:** 4433326786**Mail ID:** durgasoftonlinetraining@gmail.com**WEBSITE:** www.durgasoftonline.com**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
2.  
3. import com.durgasoft.beans.Student;  
4.  
5. public interface StudentService {  
6.     public String addStudent(Student std);  
7.     public Student searchStudent(String sid);  
8.     public String deleteStudent(String sid);  
9. }
```

**StudentServiceImpl.java**

```
1. package com.durgasoft.service;  
2.  
3. import org.springframework.beans.factory.annotation.Autowired;  
4. import org.springframework.stereotype.Service;  
5. import org.springframework.transaction.annotation.Transactional;  
6.  
7. import com.durgasoft.beans.Student;  
8. import com.durgasoft.dao.StudentDao;  
9. import com.durgasoft.entity.StudentEntity;  
10.  
11.  
12. @Service("studentService")  
13. public class StudentServiceImpl implements StudentService {  
14.  
15.     @Autowired  
16.     private StudentDao studentDao;  
17.  
18.     @Transactional  
19.     @Override  
20.     public String addStudent(Student std) {  
21.         StudentEntity stdEntity = new StudentEntity();  
22.         stdEntity.setSid(std.getSid());  
23.         stdEntity.setSname(std.getSname());  
24.         stdEntity.setSaddr(std.getSaddr());  
25.  
26.         String status = studentDao.add(stdEntity);  
27.         return status;  
28.     }  
29.  
30.     @Override  
31.     public Student searchStudent(String sid) {  
32.         StudentEntity stdEntity = studentDao.search(sid);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

33.     Student std = null;
34.     if(stdEntity == null) {
35.         std = null;
36.     }else {
37.         std = new Student();
38.         std.setSid(stdEntity.getSid());
39.         std.setSname(stdEntity.getSname());
40.         std.setSaddr(stdEntity.getSaddr());
41.     }
42.     return std;
43. }
44.
45. @Transactional
46. @Override
47. public String deleteStudent(String sid) {
48.     String status = studentDao.delete(sid);
49.     return status;
50. }
51.

```

StudentDao.java

```

1. package com.durgasoft.dao;
2.
3. import com.durgasoft.entity.StudentEntity;
4.
5. public interface StudentDao {
6.     public String add(StudentEntity stdEntity);
7.     public StudentEntity search(String sid);
8.     public String delete(String sid);
9.

```

StudentDaoImpl.java

```

1. package com.durgasoft.dao;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.springframework.beans.factory.annotation.Autowired;
6. import org.springframework.orm.hibernate4.HibernateTemplate;
7. import org.springframework.stereotype.Repository;
8. import org.springframework.transaction.annotation.Propagation;
9. import org.springframework.transaction.annotation.Transactional;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10.
11. import com.durgasoft.entity.StudentEntity;
12.
13. @Repository("studentDao")
14.
15. public class StudentDaoImpl implements StudentDao {
16.
17.
18.
19.     @Autowired
20.     private HibernateTemplate hibernateTemplate;
21.     String status = "";
22.
23.     @Override
24.     public String add(StudentEntity stdEntity) {
25.         try {
26.
27.             StudentEntity std = (StudentEntity) hibernateTemplate.get(StudentEntity.class, stdEntity.getId());
28.             if(std == null) {
29.                 String pk_Val = (String) hibernateTemplate.save(stdEntity);
30.                 if(pk_Val.equals(stdEntity.getId())) {
31.                     status = "Student Inserted Successfully";
32.                 }else {
33.                     status = "Student Insertion Failure";
34.                 }
35.             }else {
36.                 status = "Student Existed Already";
37.             }
38.         } catch (Exception e) {
39.             status = "Student Insertion Failure";
40.             e.printStackTrace();
41.         }
42.         return status;
43.     }
44.
45.     @Override
46.     public StudentEntity search(String sid) {
47.         StudentEntity stdEntity = null;
48.         try {
49.             stdEntity = (StudentEntity) hibernateTemplate.get(StudentEntity.class, sid);
50.
51.         } catch (Exception e) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

52.         e.printStackTrace();
53.     }
54.     return stdEntity;
55. }
56.
57. @Override
58. public String delete(String sid) {
59.     try {
60.         StudentEntity stdEntity = hibernateTemplate.get(StudentEntity.class, sid);
61.         if(stdEntity == null) {
62.             status = "Student Not Existed";
63.         }else {
64.             hibernateTemplate.delete(stdEntity);
65.             status = "Student Deleted SUccessfully";
66.         }
67.     } catch (Exception e) {
68.         e.printStackTrace();
69.         status = "Student Deletion Failure";
70.     }
71.     return status;
72. }
73.
74. }
```

Student.java

```

1. package com.durgasoft.beans;
2.
3. public class Student {
4.     private String sid;
5.     private String sname;
6.     private String saddr;
7.
8.     public String getSid() {
9.         return sid;
10.    }
11.    public void setSid(String sid) {
12.        this.sid = sid;
13.    }
14.    public String getSname() {
15.        return sname;
16.    }
17.    public void setSname(String sname) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
18.     this.sname = sname;
19. }
20. public String getSaddr() {
21.     return saddr;
22. }
23. public void setSaddr(String saddr) {
24.     this.saddr = saddr;
25. }
26.
27.
28.}
```

StudentEntity.java

```
1. package com.durgasoft.entity;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="student")
10. public class StudentEntity {
11.     @Id
12.     @Column(name="SID")
13.     private String sid;
14.     @Column(name="SNAME")
15.     private String sname;
16.     @Column(name="SADDR")
17.     private String saddr;
18.
19.     public String getSid() {
20.         return sid;
21.     }
22.     public void setSid(String sid) {
23.         this.sid = sid;
24.     }
25.     public String getSname() {
26.         return sname;
27.     }
28.     public void setSname(String sname) {
29.         this.sname = sname;
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
30. }
31. public String getSaddr() {
32.     return saddr;
33. }
34. public void setSaddr(String saddr) {
35.     this.saddr = saddr;
36. }
37.}
```

**tiles-defs.xml**

```
1. <?xml version="1.0" encoding="UTF-8"?>
2.
3. <!DOCTYPE tiles-definitions PUBLIC "-
//Apache Software Foundation//DTD Tiles Configuration 2.1//EN"
4. "http://tiles.apache.org/dtds/tiles-config_2_1.dtd">
5.
6. <tiles-definitions>
7.     <definition name="welcomeDef" template="/WEB-INF/layout.jsp">
8.         <put-attribute name="header" value="/WEB-INF/header.jsp"/>
9.         <put-attribute name="menu" value="/WEB-INF/menu.jsp"/>
10.        <put-attribute name="body" value="/WEB-INF/welcome.jsp"/>
11.        <put-attribute name="footer" value="/WEB-INF/footer.jsp"/>
12.    </definition>
13.    <definition name="addDef" extends="welcomeDef">
14.        <put-attribute name="body" value="/WEB-INF/addstudent.jsp"/>
15.    </definition>
16.    <definition name="searchDef" extends="welcomeDef">
17.        <put-attribute name="body" value="/WEB-INF/searchstudent.jsp"/>
18.    </definition>
19.    <definition name="deleteDef" extends="welcomeDef">
20.        <put-attribute name="body" value="/WEB-INF/deletetestudent.jsp"/>
21.    </definition>
22.    <definition name="statusDef" extends="welcomeDef">
23.        <put-attribute name="body" value="/WEB-INF/status.jsp"/>
24.    </definition>
25.    <definition name="studentDetailsDef" extends="welcomeDef">
26.        <put-attribute name="body" value="/WEB-INF/studentdetails.jsp"/>
27.    </definition>
28. </tiles-definitions>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

ds-servlet.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xmlns:p="http://www.springframework.org/schema/p"
5.   xmlns:context="http://www.springframework.org/schema/context"
6.   xmlns:aop="http://www.springframework.org/schema/aop"
7.   xmlns:tx="http://www.springframework.org/schema/tx"
8.
9.   xsi:schemaLocation="
10.     http://www.springframework.org/schema/beans
11.     http://www.springframework.org/schema/beans/spring-beans.xsd
12.     http://www.springframework.org/schema/context
13.     http://www.springframework.org/schema/context/spring-context.xsd
14.     http://www.springframework.org/schema/tx
15.     http://www.springframework.org/schema/tx/spring-tx.xsd
16.     http://www.springframework.org/schema/aop
17.     http://www.springframework.org/schema/aop/spring-aop.xsd">
18.
19. <context:component-scan base-package="com.durgasoft" />
20. <tx:annotation-driven transaction-manager="hibernateTransactionManager" />
21.
22. <bean id="viewResolver"
23.   class="org.springframework.web.servlet.view.UrlBasedViewResolver">
24.   <property name="viewClass">
25.     <value>
26.       org.springframework.web.servlet.view.tiles2.TilesView
27.     </value>
28.   </property>
29. </bean>
30.
31. <bean id="tilesConfigurer"
32.   class="org.springframework.web.servlet.view.tiles2.TilesConfigurer">
33.   <property name="definitions">
34.     <list>
35.       <value>/WEB-INF/tiles-defs.xml</value>
36.     </list>
37.   </property>
38. </bean>
39.
40. <bean id="dataSource"
41.   class="org.springframework.jdbc.datasource.DriverManagerDataSource">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

42.   <property name="driverClassName" value="oracle.jdbc.OracleDriver"/>
43.   <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"/>
44.   <property name="username" value="system"/>
45.   <property name="password" value="durga"/>
46. </bean>
47.
48. <bean id="sessionFactory"
49.   class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">
50.   <property name="dataSource" ref="dataSource"/>
51.   <property name="annotatedClasses">
52.     <list>
53.       <value>com.durgasoft.entity.StudentEntity</value>
54.     </list>
55.   </property>
56.   <property name="hibernateProperties">
57.     <props>
58.       <prop key="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</prop>
59.       <!-- <prop key="hibernate.show_sql">true</prop> -->
60.     </props>
61.   </property>
62. </bean>
63.
64. <bean id="hibernateTransactionManager"
65.   class="org.springframework.orm.hibernate4.HibernateTransactionManager">
66.   <property name="sessionFactory" ref="sessionFactory"/>
67. </bean>
68.
69. <bean id="hibernateTemplate" class="org.springframework.orm.hibernate4.HibernateTe
mplate">
70.   <property name="sessionFactory" ref="sessionFactory"/>
71. </bean>
72.
73. </beans>

```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>tilesapp</display-name>
4.   <welcome-file-list>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. <welcome-file>index.html</welcome-file>
6. <welcome-file>index.htm</welcome-file>
7. <welcome-file>index.jsp</welcome-file>
8. <welcome-file>default.html</welcome-file>
9. <welcome-file>default.htm</welcome-file>
10. <welcome-file>default.jsp</welcome-file>
11. </welcome-file-list>
12.
13. <servlet>
14.   <servlet-name>ds</servlet-name>
15.   <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
16.   <load-on-startup>1</load-on-startup>
17. </servlet>
18.
19. <servlet-mapping>
20.   <servlet-name>ds</servlet-name>
21.   <url-pattern>/</url-pattern>
22. </servlet-mapping>
23.
24. </web-app>
```

To run this application we have to use the following JARs in web application lib folder.

- 1) Spring MVC + Spring ORM + Spring Tx + Spring JDBC + Spring AOP
- 2) Tiles Jars and its supporting jars
- 3) Jstl jars If we use JstlView class.
- 4) Hibernate Jars
- 5) ojdbc6.jar
- 6) Commons bean utils+commons digester + log4j + slf4j-log4j12 + slf4j

Spring JARS:

- + spring-aop-4.3.9.RELEASE.jar
- + spring-aspects-4.3.9.RELEASE.jar
- + spring-beans-4.3.9.RELEASE.jar
- + spring-context-4.3.9.RELEASE.jar
- + spring-context-support-4.3.9.RELEASE.jar
- + spring-core-4.3.9.RELEASE.jar
- + spring-expression-4.3.9.RELEASE.jar
- + spring-jdbc-4.3.9.RELEASE.jar
- + spring-orm-4.3.9.RELEASE.jar
- + spring-tx-4.3.9.RELEASE.jar
- + spring-web-4.3.9.RELEASE.jar
- + spring-webmvc-4.3.9.RELEASE.jar

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Tiles and dependent JARS:

- + commons-beanutils-1.8.3.jar
- + commons-digester-2.1.jar
- + log4j.jar
- + slf4j-log4j12.jar
- + slf4j.jar
- + tiles-api-2.2.2.jar
- + tiles-core-2.2.2.jar
- + tiles-jsp-2.2.2.jar
- + tiles-servlet-2.2.2.jar
- + tiles-template-2.2.2.jar

Jstl Jars

- + taglibs-standard-impl-1.2.5.jar
- + taglibs-standard-spec-1.2.5.jar

Hibernate Jars:

- + antlr-2.7.7.jar
- + commons-logging-1.2.jar
- + dom4j-1.6.1.jar
- + hibernate-commons-annotations-4.0.5.Final.jar
- + hibernate-core-4.3.11.Final.jar
- + hibernate-entitymanager-4.3.11.Final.jar
- + hibernate-jpa-2.1-api-1.0.0.Final.jar
- + jandex-1.1.0.Final.jar
- + javassist-3.18.1-GA.jar
- + jboss-logging-3.1.3.GA.jar
- + jboss-logging-annotations-1.2.0.Beta1.jar
- + jboss-transaction-api_1.2_spec-1.0.0.Final.jar
- + ojdbc6.jar

ALL Jars Together

- + antlr-2.7.7.jar
- + commons-beanutils-1.8.3.jar
- + commons-digester-2.1.jar
- + commons-logging-1.2.jar
- + dom4j-1.6.1.jar
- + hibernate-commons-annotations-4.0.5.Final.jar
- + hibernate-core-4.3.11.Final.jar
- + hibernate-entitymanager-4.3.11.Final.jar
- + hibernate-jpa-2.1-api-1.0.0.Final.jar
- + jandex-1.1.0.Final.jar

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ⊕ javassist-3.18.1-GA.jar
- ⊕ jboss-logging-3.1.3.GA.jar
- ⊕ jboss-logging-annotations-1.2.0.Beta1.jar
- ⊕ jboss-transaction-api_1.2_spec-1.0.0.Final.jar
- ⊕ log4j.jar
- ⊕ ojdbc6.jar
- ⊕ slf4j-log4j12.jar
- ⊕ slf4j.jar
- ⊕ spring-aop-4.3.9.RELEASE.jar
- ⊕ spring-aspects-4.3.9.RELEASE.jar
- ⊕ spring-beans-4.3.9.RELEASE.jar
- ⊕ spring-context-4.3.9.RELEASE.jar
- ⊕ spring-context-support-4.3.9.RELEASE.jar
- ⊕ spring-core-4.3.9.RELEASE.jar
- ⊕ spring-expression-4.3.9.RELEASE.jar
- ⊕ spring-jdbc-4.3.9.RELEASE.jar
- ⊕ spring-orm-4.3.9.RELEASE.jar
- ⊕ spring-tx-4.3.9.RELEASE.jar
- ⊕ spring-web-4.3.9.RELEASE.jar
- ⊕ spring-webmvc-4.3.9.RELEASE.jar
- ⊕ taglibs-standard-impl-1.2.5.jar
- ⊕ taglibs-standard-spec-1.2.5.jar
- ⊕ tiles-api-2.2.2.jar
- ⊕ tiles-core-2.2.2.jar
- ⊕ tiles-jsp-2.2.2.jar
- ⊕ tiles-servlet-2.2.2.jar
- ⊕ tiles-template-2.2.2.jar

Spring_Struts Integration:

Struts Overview:

- ❖ Framework is a semi implemented application, it will be used to design applications in simplified manner.
- ❖ Framework is prefabricated software units that programmer can share, customize, reuse in order to simplify application development.
- ❖ Framework is the collection of predefined classes and interfaces to simplify application development.

In general, in Applications development, Frameworks will provide the following advantages.

1. Frameworks will provide common implementation in all the projects as predefined implementation

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX: Controller Servlets and the services like I18N, Exception Handling, Validations ,.....

2. Frameworks will provide standard template to design applications.
3. Frameworks will provide standard flow of execution between the components.
4. Frameworks will reduce application development time.
5. Frameworks will reduce application development cost.
6. Frameworks will improve productivity.
7. Frameworks will provide modularity in application development.

There are two types of Frameworks.

1. Web Frameworks
2. Application Frameworks

What is the difference between web frameworks and Application Frameworks?

Ans:

Web Frameworks will provide very good environment to prepare and execute web applications only.

EX: Struts, JSF.

Application Frameworks will provide very good environment to prepare and execute any type of JAVA/J2EE Applications including Standalone Applications, Web applications, Distributed Applications,

EX: Spring

Struts is MVC based web framework provided by Apache Software Foundations.

Struts is existed in two versions.

1. Struts1.x
2. Struts2.x

If we want to prepare Struts applications in Struts1.x version then we have to use the following components

1. View
2. Deployment Descriptor
3. Controller [ActionServlet]
4. Action class
5. Action Form
6. Struts Configuration File

1. View:

In Struts applications, View part is representing presentation part.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In web applications, there are two types of presentation part.

1. Informational Presentation part
2. Form Based Presentation part

1. Informational Presentation part

This type view part is able to provide only information to the user, which includes status of the server side actions like Success, failure, ... and display a particular database table data,.....

EXAMPLE: success.html

```
<h1> Login Success </h1>
```

2. Form Based Presentation part:

This type of view part is able to provide a form to collect data from users and to submit data to the server side applications.

In Struts applications, to prepare Presentation part we are able to use plain HTML tags, but, which are not suggestible. In Struts applications it is suggestible to use Struts provided tag library.

EX: loginform.html

```
1.      <html>
2.  <body>
3.  <form method="POST" action="login.*">
4.  <table>
5.  <tr>
6.    <td>User Name</td>
7.    <td><input type="text" name="uname"/></td>
8.  </tr>
9.  <tr>
10.   <td>Password</td>
11.   <td><input type="password" name="upwd"/></td>
12. </tr>
13. <tr>
14.   <td><input type="submit" value="Login"/></td>
15. </tr>
16. </table></form></body></html>
```

The above login form with Struts provided html tag library

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX: loginform.jsp

```
1. <%@taglib uri="http://struts.apache.org/tags-html" prefix="html" %>
2. <html:html>
3. <body>
4. <html:form method="POST" action="login.do">
5. <table>
6. <tr>
7. <td>User Name</td>
8. <td><html:text property="uname"/></td>
9. </tr>
10. <tr>
11. <td>Password</td>
12. <td><html:password property="upwd"/></td>
13. </tr>
14. <tr>
15. <td><html:submit>Login</html:submit></td>
16. </tr>
17. </table></html:form></body></html>
```

2. Deployment Descriptor:

Deployment descriptor is web.xml file, it will provide description about our web application which is required by the container in order to perform server side actions.

In general, in web applications, web.xml file will provide the following configuration details.

1. Display names configurations.
2. Welcome Files Configurations.
3. Servlets configurations
4. Filters Configurations
5. Listeners Configurations.
6. Session time out configurations
7. Error Pages Configurations
8. Taglib configurations

In Struts based web applications, the main intention of web.xml file is to configure controller Servlet that is ActionServlet.

EX:

```
1. <web-app>
2. <servlet>
3. <servlet-name>actionServlet</servlet-name>
4. <servlet-class>org.apache.struts.action.ActionServlet
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
5. </servlet-class>
6. <load-on-startup>1</load-on-startup>
7. </servlet>
8. <servlet-mapping>
9. <servlet-name>actionServlet</servlet-name>
10. <url-pattern>*.do</url-pattern>
11. </servlet-mapping>
12. </web-app>
```

3. Controller [ActionServlet]

In Struts Framework, ActionServlet is predefined Controller, it was provided in the form of org.apache.struts.action.ActionServlet .

In Struts applications, ActionServlet will perform the following actions.

1. ActionServlet will take request from Client.
2. ActionServlet will identify the names and locations of the ActionForm and Action class through Struts configuration file.
3. ActionServlet will load , instantiate ActionForm class.
4. ActionServlet will store form data in ActionForm object.
5. ActionServlet will load and instantiation Action class.
6. ActionServlet will execute execute(--) method in Action class.
7. ActionServlet will identify view page through Struts configuration file.
8. ActionServlet will forward request to view page inorder to generate response.

Note: In Struts applications, ActionServlet is performing all the above actions by using "RequestProcessor" internally.

4. Action Class or Controller Component:

In Struts based web applications, the main intention of ActionForm or FormBean component is to manage a particular form data at Server side inorder to perform Server side data validations, to transfer data from Controller layer to model layer or Vie layer,.....

To prepare Form Bean components in Struts applications we have to use the following rules and regulations.

1. Declare an user defined class, it must be extended from org.apache.struts.action.ActionForm .
2. Form Bean class must be public, it must not be abstract and final.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. In Form Bean class, we must declare properties with the same names of the form properties.
4. In Form Bean class we must declare all properties as private and all behaviours as public.
5. In FormBean class , we must provide a separate set of setXXX() and getXXX() methods for each and every property.
6. In FormBean class, if we want to provide any constructor then we can provide constructor but that constructor must be public and 0-arg constructor.
7. In FormBean class, we can override equals(-) method and hashCode() method as per the requirement.

EXAMPLE:**LoginActionForm.java**

```

1. public class LoginActionForm extends org.apache.struts.action.ActionForm{
2.   private String uname;
3.   private String upwd;
4.   public void setUserName(String uname){
5.     this.uname = uname;
6.   }
7.   public void setUpwd(String upwd){
8.     this.upwd = upwd;
9.   }
10.  public String getUserName(){
11.    return uname;
12.  }
13.  public String setUpwd(){
14.    return upwd;
15.  }
16.}
```

5. Action Class or Controller Component:

In Struts based web applications, the main intention of Action class is to manage application logic which we want to execute by getting request from client.

In Struts based web applications, to prepare Action class we have to use the following steps.

1. Declare an user defined class and it must be extended from org.apache.struts.action.Action class.
2. Action class must be public class and it must not be abstract class.
3. In Action class we must override either of the following methods.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest  
request, HttpServletResponse response) throws Exception
```

```
public ActionForward execute(ActionMapping mapping, ActionForm form, ServletRequest  
request, ServletResponse response) throws Exception.
```

Note: In execute() method we must return ActionForward object with a particular Forward key inorder to identify target view page, for this, we have to use the following method from ActionMapping.

```
public ActionForward findForward(String key)
```

EX: LoginAction.java

```
1. public class LoginAction extends org.apache.struts.action.Action{  
2.   public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest  
       request, HttpServletResponse response) throws Exception{  
3.     ----Appl logic----  
4.     String status = "success"/"failure";  
5.     return mapping.findForward(status);  
6.   }  
7. }
```

6. Struts Configuration File:

struts-config.xml

- In Struts based web applications, the main intention of struts configuration file is to provide mappings between user forms and the respective ActionForm classes and Action classes ,..... which are required by the ActionServlet inorder to perform Server side actions.
- In Struts based web applications, the default name and location of configuration file is "struts-config.xml" and "WEB-INF" location, but, it is possible to change this default name and location but we must give that new name and location to the Struts Framework.
- In Struts based web applications, configuration file is able to provide the following configurations.
 1. Datasource configurations.
 2. Global Forwards configurations.
 3. Form Beans Configurations.
 4. Action classes configurations.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

5. Message Resources configurations.
6. Controller configurations.
7. Plugin configurations
8. Global Exceptions configurations.

To prepare basic Struts based web applications we need to provide ActionForm class configuration and action class configuration in struts configuration file.

```
1. <!DOCTYPE ---->
2. <struts-config>
3.   <form-beans>
4.     <form-bean name="--" type="--"/>
5.     -----
6.   </form-beans>
7.   <action-mappings>
8.     <action path="/---" name="--" type="--">
9.       <forward name="--" path="/---"/>
10.      -----
11.     </action>
12.     -----
13.   </action-mappings>
14.   -----
15. </struts-config>
```

- ✓ Where <struts-config> is a root tag, it will include struts application configuration details.
- ✓ Where <form-beans> tag is able to include no of form beans configurations.
- ✓ Where <form-bean> tag is able to provide single form bean class configuration.
- ✓ Where "name" attribute in <form-bean> tag is able to provide logical name to Form Bean component.
- ✓ Where "type" attribute in <form-bean> tag will take fully qualified name of the respective form bean class.
- ✓ Where <action-mappings> tag is able to include no of actions configurations.
- ✓ Where <action> tag is able to provide mapping between user form, ActionForm class and the respective Action class.
- ✓ Where "path" attribute in <action> tag is able to take url pattern which we specified in User form.
- ✓ Where "name" attribute in <action> tag will take logical name of the form bean class which we specified in struts configuration file under form beans configurations.
- ✓ Where "type" attribute in <action> tag is able to take fully qualified name of the Action class.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ Where <forward> tag is able to provide mapping between forward key which is returned from execute() method in Action class and the target view page.
- ✓ Where "name" attribute in <forward> tag is able to take forward key.
- ✓ Where "path" attribute in <forward> tag is able to take the name and location of target view page.

EX:

```

1. <!DOCTYPE ..... >
2. <struts-config>
3.   <form-beans>
4.     <form-
      bean name="loginForm"           type="com.durgasoft.beans.LoginActionForm"/>
6.   </form-beans>
7.   <action-mappings>
8.     <action path="/login" name="loginForm"           type="com.durgasoft.action.LoginAct
      ion">
9.       <forward name="success" path="/success.html"/>
10.      <forward name="failure" path="/failure.html"/>
11.   </action-mappings>
12. </struts-config>

```

Steps:

- 
1. Download Struts JARs from internet.
 2. Create Dynamic Project in Eclipse.
 3. Copy all Struts related JARs in lib folder in dynamic web project.
 4. Configure ActionServlet in web.xml file.
 5. Prepare User forms
 6. Prepare Form Bean class.
 7. Create Action class.
 8. Create Struts configuration file.
 9. Run dynamic web application.

Example (struts Application)**loginform.html**

```

1. <!DOCTYPE html>
2. <html>
3.   <head>
4.     <meta charset="ISO-8859-1">
5.     <title>Insert title here</title>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Form</h3>
10. <form method="POST" action="login.do">
11. <table>
12. <tr>
13.   <td>User Name</td>
14.   <td><input type="text" name="uname"/></td>
15. </tr>
16. <tr>
17.   <td>Password</td>
18.   <td><input type="password" name="upwd"/></td>
19. </tr>
20. <tr>
21.   <td><input type="submit" value="Login"/></td>
22. </tr>
23. </table>
24. </form>
25. </body>
26. </html>
```

success.html

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status</h3>
10. <font color="red" size="6">
11. <b>User Login Success</b>
12. </font>
13. <h5>
14. <a href=".//loginform.html">|User Login Form|</a>
15. </h5>
16. </body>
17. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Failure.html

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status</h3>
10. <font color="red" size="6">
11. <b>User Login Failure</b>
12. </font>
13. <h5>
14. <a href=".//loginform.html">|User Login Form|</a>
15. </h5>
16. </body>
17. </html>
```

LoginActionForm.java

```
1. package com.durgasoft.beans;
2.
3. import org.apache.struts.action.ActionForm;
4.
5. public class LoginActionForm extends ActionForm {
6.     private String uname;
7.     private String upwd;
8.
9.     public String getUname() {
10.         return uname;
11.     }
12.     public void setUname(String uname) {
13.         this.uname = uname;
14.     }
15.     public String getUpwd() {
16.         return upwd;
17.     }
18.     public void setUpwd(String upwd) {
19.         this.upwd = upwd;
20.     }
21. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

LoginAction.java

```

1. package com.durgasoft.action;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.apache.struts.action.Action;
7. import org.apache.struts.action.ActionForm;
8. import org.apache.struts.action.ActionForward;
9. import org.apache.struts.action.ActionMapping;
10.
11. import com.durgasoft.beans.LoginActionForm;
12.
13. public class LoginAction extends Action {
14.     @Override
15.     public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request,
16.             HttpServletResponse response) throws Exception {
17.         LoginActionForm laf = (LoginActionForm)form;
18.         String uname = laf.getUname();
19.         String upwd = laf.getUpwd();
20.         String status = "";
21.         if(uname.equals("durga") && upwd.equals("durga")) {
22.             status = "success";
23.         }else {
24.             status = "failure";
25.         }
26.         return mapping.findForward(status);
27.     }
28. }
```

struts-config.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE struts-config PUBLIC
3.     "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
4.     "http://struts.apache.org/dtds/struts-config_1_3.dtd">
5. <struts-config>
6.     <form-beans>
7.         <form-bean name="loginForm" type="com.durgasoft.beans.LoginActionForm"/>
8.     </form-beans>
9.     <action-mappings>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10.   <action path="/login" name="loginForm" type="com.durgasoft.action.LoginAction">
11.     <forward name="success" path="/success.html"/>
12.     <forward name="failure" path="/failure.html"/>
13.   </action>
14. </action-mappings>
15.</struts-config>
```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.c
   om/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
   app_2_5.xsd" id="WebApp_ID" version="2.5">
3.   <display-name>loginapp</display-name>
4.   <welcome-file-list>
5.     <welcome-file>loginform.html</welcome-file>
6.   </welcome-file-list>
7.   <servlet>
8.     <servlet-name>actionServlet</servlet-name>
9.     <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
10.    <load-on-startup>1</load-on-startup>
11.  </servlet>
12.  <servlet-mapping>
13.    <servlet-name>actionServlet</servlet-name>
14.    <url-pattern>*.do</url-pattern>
15.  </servlet-mapping>
16.</web-app>
```

If we want to integrate Struts application with Spring Framework then we have to use the following steps.

1. Prepare User Forms and View part.
2. Prepare ActionForm class.
3. Prepare Action class.
4. Prepare Business class.
5. Prepare Struts configuration File.
6. Prepare Spring Configuration File.
7. Prepare web.xml file.

1. Prepare User Forms and View part:

In Struts and Spring Integration applications, we have to prepare presentation part as per Struts rules and regulations only.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

To prepare User Interface in Struts applications we will use either normal html tags or we will use Struts provided tag library.

EX:

```
<form method="POST" action="login.do">  
User Name<input type="text" name="uname"/><br>  
Password<input type="password" name="upwd"/><br>  
<input type="submit" value="Login"/>  
</form>
```

success.jsp

```
<h1> User Login Success</h1>
```

failure.jsp

```
<h1> User Login Failure </h1>
```

2. Prepare ActionForm class.

In Struts based web applications, we have to prepare ActionForm class inorder to manage user form data at Server side .

In Struts applications, to prepare ActionForm class we will use the following steps.

1. Declare an USer defined class.
2. Extend org.apache.struts.action.ActionForm class to user defined class.
3. Declare properties as per user form in ActionForm class and provide setXXX() and getXXX() methods for each and every property.

EX:

```
public class LoginActionForm extends ActionForm{  
private String uname;  
private String upwd;  
setXXX() and getXXX()  
}
```

3. Prepare Action class.

In Struts based web applications, the main intention of Action class is to include business logic or to provide Business Components provided business method calls.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In Struts with Spring integration applications we will provide application business logic in Spring Bean components and it must be accessed from Struts Action classes.

To prepare Action class in Struts with Spring integration applications we have to use the following steps.

- Declare an user defined class.
- Extend org.springframework.web.struts.ActionSupport class to user defined class.
- Provide application logic by overriding execute() method.
- In execute() method get ApplicationContext object by accessing getWebApplicationContext() method and get Spring Bean objects by using getBean() method.
- Access Business methods which we provided in Spring bean objects.

Note: The main intention of org.springframework.web.struts.ActionSupport class is to provide getWebApplicationContext() method inorder to get ApplicationContext object inorder to get Spring provided Bean objects.

```
public ApplicationContext getWebApplicationContext()
```

EX:

```
public class LoginAction extends ActionSupport {  
    public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response) throws Exception {  
        LoginActionForm laf = (LoginActionForm)form;  
        String uname = laf.getUname();  
        String upwd = laf.getUpwd();  
        ApplicationContext context = getWebApplicatinContext();  
        UserService us =(UserService)context.getBean("userService");  
        String status = us.checkLogin(uname, upwd);  
        return mapping.findForward(status);  
    }  
}
```

4. Prepare Business class:

In Struts with Spring Integration application, we will provide Business component as per Spring rules and regulation.

EX:

```
public class UserService{  
    String status = "";
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public String checkLogin(String uname, String upwd){
    if(uname.equals("durga") && upwd.equals("durga")){
        status = "success";
    }else{
        status = "failure";
    }
}
```

5. Prepare Struts configuration File:

In Struts with Spring Integration applications, we have to provide all Struts configurations like FormBeans, Action classes,... and we must provide plug-in configuration with the "org.springframework.web.struts.ContextLoaderPlugIn" with the property "contextConfigLocation" inorder to provide name and location of the Spring configuration.

```
<struts-config>
    <form-beans>
        <form-bean name="loginActionForm"
type="com.durgasoft.formbeans.LoginActionForm"/>
    </form-beans>
    <action-mappings>
        <action path="/login" name="loginActionForm"
type="com.durgasoft.action.LoginAction">
            <forward name="success" path="/success.jsp"/>
            <forward name="failure" path="/failure.jsp"/>
        </action>
    </action-mappings>
    <plug-in className="org.springframework.web.struts.ContextLoaderPlugIn">
        <set-property property="contextConfigLocation" value="/WEB-
INF/applicationContext.xml" />
    </plug-in>
</struts-config>
```

6. Prepare Spring Configuration File.

In Struts with Spring Integration application we will prepare Spring configuration file with all the beans configuration.

EX:

```
<beans>
    <bean id="userService" class="com.durgasoft.service.UserService"/>
</beans>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



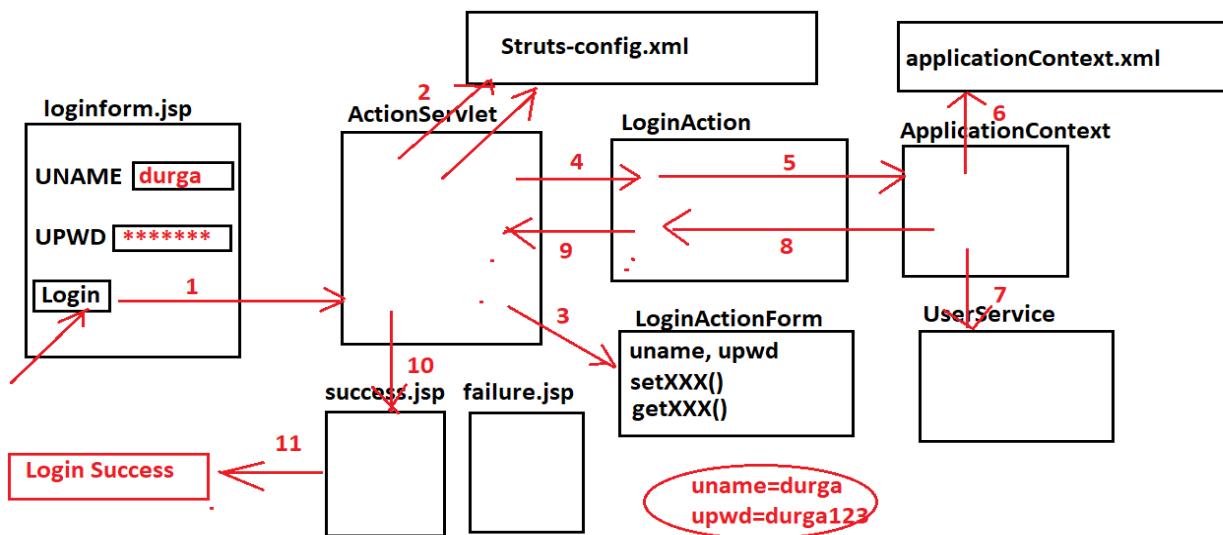
BY NAGOOR BABU

7. Prepare web.xml file.

In Struts with Spring integration application , we will provide ActionServlet Configuration in web.xml file as per Struts rules and regulations.

EX:

```
<web-app>
  <display-name>struts_spring_app</display-name>
  <welcome-file-list>
    <welcome-file>loginform.jsp</welcome-file>
  </welcome-file-list>
  <servlet>
    <servlet-name>actionServlet</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>actionServlet</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>
</web-app>
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**loginform.jsp**

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
11. <h3 style="color: blue;" align="center">User Login Page</h3>
12. <form method="POST" action="login.do">
13. <center>
14. <table>
15. <tr>
16.   <td>User Name</td>
17.   <td><input type="text" name="uname"/></td>
18. </tr>
19. <tr>
20.   <td>Password</td>
21.   <td><input type="password" name="upwd"/></td>
22. </tr>
23. <tr>
24.   <td><input type="submit" value="Login"/></td>
25. </tr>
26. </table>
27. </center>
28. </form>
29. </body>
30. </html>
```

success.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br>
11. <h1 style="color: red;" align="center">User Login Success</h1>
12. </body>
13. </html>
```

failure.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.     pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//
4. /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
5. <html>
6. <head>
7. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8. <title>Insert title here</title>
9. </head>
10. <br><br>
11. <h1 style="color: red;" align="center">User Login Failure</h1>
12. </body>
13. </html>
```

LoginActionForm.java

```

1. package com.durgasoft.formbeans;
2.
3. import org.apache.struts.action.ActionForm;
4.
5. public class LoginActionForm extends ActionForm {
6.     private String uname;
7.     private String upwd;
8.
9.     public String getUsername() {
10.         return uname;
11.     }
12.     public void setUsername(String uname) {
13.         this.uname = uname;
14.     }
15. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14. }
15. public String getUpwd() {
16.     return upwd;
17. }
18. public void setUpwd(String upwd) {
19.     this.upwd = upwd;
20. }
21.
22.
23.}
```

LoginAction.java

```
1. package com.durgasoft.action;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.apache.struts.action.ActionForm;
7. import org.apache.struts.action.ActionForward;
8. import org.apache.struts.action.ActionMapping;
9. import org.springframework.web.struts.ActionSupport;
10.
11. import com.durgasoft.formbeans.LoginActionForm;
12. import com.durgasoft.service.UserService;
13.
14. public class LoginAction extends ActionSupport {
15.
16.
17.
18.     @Override
19.     public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request,
20.         HttpServletResponse response) throws Exception {
21.         LoginActionForm laf = (LoginActionForm)form;
22.         String uname = laf.getUname();
23.         String upwd = laf.getUpwd();
24.         UserService userService = (UserService) getWebApplicationContext().getBean("userService");
25.         String status = userService.checkLogin(uname, upwd);
26.         return mapping.findForward(status);
27.     }
28. }
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

UserService.java

```
1. package com.durgasoft.service;
2.
3. public class UserService {
4.     String status = "";
5.     public String checkLogin(String uname, String upwd) {
6.         if(uname.equals("durga") && upwd.equals("durga")) {
7.             status = "success";
8.         }else {
9.             status = "failure";
10.        }
11.        return status;
12.    }
13.}
```

struts-config.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE struts-config PUBLIC
3.     "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
4.     "http://struts.apache.org/dtds/struts-config_1_3.dtd">
5. <struts-config>
6.     <form-beans>
7.         <form-
8.             bean name="loginActionForm" type="com.durgasoft.formbeans.LoginActionForm"/>
9.         </form-beans>
10.        <action-mappings>
11.            <action path="/login" name="loginActionForm" type="com.durgasoft.action.LoginAction">
12.                <forward name="success" path="/success.jsp"/>
13.                <forward name="failure" path="/failure.jsp"/>
14.            </action>
15.        </action-mappings>
16.        <plug-in className="org.springframework.web.struts.ContextLoaderPlugIn">
17.            <set-property property="contextConfigLocation" value="/WEB-INF/applicationContext.xml" />
18.        </plug-in>
</struts-config>
```

applicationContext.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
2. <!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
3.   "http://www.springframework.org/dtd/spring-beans.dtd">
4. <beans>
5.   <bean id="userService" class="com.durgasoft.service.UserService"/>
6. </beans>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3.   <display-name>struts_spring_app</display-name>
4.   <welcome-file-list>
5.     <welcome-file>loginform.jsp</welcome-file>
6.   </welcome-file-list>
7.   <servlet>
8.     <servlet-name>actionServlet</servlet-name>
9.     <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
10.    <load-on-startup>1</load-on-startup>
11.   </servlet>
12.   <servlet-mapping>
13.     <servlet-name>actionServlet</servlet-name>
14.     <url-pattern>*.do</url-pattern>
15.   </servlet-mapping>
16. </web-app>
```

To run this application we have to use the following JARs.

- Struts1.3.10 Jars + Spring2.5 jars

- + antlr-2.7.2.jar
- + bsf-2.3.0.jar
- + commons-beanutils-1.8.0.jar
- + commons-chain-1.2.jar
- + commons-digester-1.8.jar
- + commons-fileupload-1.1.1.jar
- + commons-io-1.1.jar
- + commons-logging-1.0.4.jar
- + commons-validator-1.3.1.jar
- + jstl-1.0.2.jar
- + oro-2.0.8.jar
- + spring-aop.jar

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- + spring-beans.jar
- + spring-context-support.jar
- + spring-context.jar
- + spring-core.jar
- + spring-jdbc.jar
- + spring-jms.jar
- + spring-orm.jar
- + spring-test.jar
- + spring-tx.jar
- + spring-web.jar
- + spring-webmvc-portlet.jar
- + spring-webmvc-struts.jar
- + spring-webmvc.jar
- + standard-1.0.6.jar
- + struts-core-1.3.10.jar
- + struts-el-1.3.10.jar
- + struts-extras-1.3.10.jar
- + struts-faces-1.3.10.jar
- + struts-mailreader-dao-1.3.10.jar
- + struts-scripting-1.3.10.jar
- + struts-taglib-1.3.10.jar
- + struts-tiles-1.3.10.jar

SPRING-JSF Integration:

JSF Introduction:

- JSF stands for Java Server Faces, it is MVC based web framework, it can be used to prepare web applications in standard mode.
- JSF is MVC based web Framework, it has very good focus on Presentation layer in enterprise application development.
- JSF is initially provided by JCP[Java Community Process], later on it was given to SUN Microsystems.
- JSF is provided by SUN Microsystems in the form of the following versions.
 1. JSF1.X
 2. JSF2.X
 3. JSF3.X

JSF Features:

- JSF is view layered Framework, it has very good focus on view layer in MVC based applications.
- JSF is component based framework or Technology, it has very good component tag library to prepare web applications.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- JSF is having very good GUI tag library to prepare web applications with very good look and feel, JSF is providing very good environment to prepare our own custom tags inorder to prepare our own components.
- JSF is having both client side user interface and Server side user interface, it able to manage User interface at server side by preparing Component Tree at Server side.
- JSF is having Statefull GUI component, they are able to manage their state at Server side.
- JSF is having very good Validations Support, it able to allow to define our own Custom Validators.
- JSF is having very good Convertors support, it able to allow to define own convertors.
- JSF is having very good Internationalization support inorder prepare web applications w.r.t the Local users
- JSF is having very good Renderring mechanisms to generate response in different formats as per the requirements, it allows us to prepare our own renderring mechanisms.
- JSF is having very good Event-Notification model to implement Event Handling, it allows our own custom Events in web applications.
- JSF is providing very good environment to use Third part vendors provided GUI Components libraries like primefaces, richfaces, ICEFaces,....

JSF Components:

To prepare web applications by using JSF then we have to use the following JSF Components.

1. View / Presentation Part
2. Controller or FacesServlet
3. Managed Bean
4. Faces Configuration File
5. Deployment Descriptor or web.xml

1. View / Presentation Part:

The main intention of View part in web applications is,

1. To improve Look And Feel to the web applications.
2. To Get Starting point to the web applications inorder to access.
3. To get data from users inorder to submit to the Server side application.
4. To perform client side data validations by using Java script functions.
5. To submit different request types like GET, POST, HEAD,... from client.

In web application , to prepare view part, we will use the technologies like HTML, JSP, Velocity, Freemarker,.....

There are two types of View part we are able to use in web applications.

1. Information View
2. Form based View

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where informational view is able to display information to the user, it will include any form.

EX: status.html, success.html, failure.html,....

Where form based view part is able to include user forms to get data from users and to submit data to server side applications.

In JSF based web applications, to prepare View part we have to use JSF provided tag library.

To prepare View part , JSF has provided the following two types of tags.

1. Core Tags
2. Html Tags

Where core tag library includes the tags which are used for View representations, Conversions, Validatins,....

EX:

```
<f:convertNumber/>
<f:convertDateTime/>
<f:validateLength/>
<f:validateLongRange/>
```

Where Html tag library include the tags which are reperesenting Html Components

EX:

```
<h:outputText/>
<h:inputText/>
<h:inputSecret/>
<h:inputTextArea/>
```

In JSP based web applications, to prepare Usaer form we have to use the following steps.

1. Declare view part by using <f:view> tag.
2. Include Html header part and body part in <f:view> tag.
3. Prepare user form by using <h:form> tag.
4. Prepare Panelgrid by using <h:panelGrid> tag inorder to prepare layout for GUI component.
5. Prepare html components by using the following html tags.

<h:outputText> ---> Label
<h:inputText> ----> Text Field.
<h:inputSecret> --> Password field.
<h:commandButton/>--> Button.

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Controller or FacesServlet

In General, in web applications, the main intention of controller is,

1. Getting request from Client.
2. Identifying Model component.
3. Instantiating Model component and executing Business logic.
4. Identifying View part.
5. Forwarding request to View part.

In JSF Based web applications, FacesServlet is acting as controller, it was provided by JSF in the form of a predefined class like "javax.faces.webapp.FacesServlet".

In JSF based web applications, FacesServlet will get Request from client and it will process the request by performing the following request processing lifecycle actions.

1. Restore View
2. Apply Request values.
3. Process Validations.
4. Update Model Values.
5. Invoke Application.
6. Render Response.

3. Managed Bean:

In MVC based applications, the main intention of Bean classes is,

1. To manage User form Data.
2. To implement Data validations.
3. To implement Application Business Logic,....

In JSF Based web applications, Java Bean class is called as Managed Bean class, it is acting as model component.

IN JSF Based web application, To prepare Managed Beans, we have to use the following rules and regulations.

1. Managed Bean class must be a POJO class, it must not be extended and implemented any predefined library.
2. It will take all properties as per User forms.
3. It will include a separate set of SetXXX() and getXXX() methods for each and every property.
4. It will include business methods which are bounded with action attribute in CommandButton in User forms.
5. If we want to provide constructor in Managed Bean class then provide constructor, it must be public and 0- arg.
6. If we want to provide our own comparisons between managed Bean objects then we must override equals() method

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

7. If we want to generate Hashcode values to Managed Bean objects then we have to override hashCode() method.

Example:

```

1. package com.durgasoft.beans;
2.
3. public class LoginBean {
4.     private String uname;
5.     private String upwd;
6.
7.     public String getUsername() {
8.         return uname;
9.     }
10.    public void setUsername(String uname) {
11.        this.uname = uname;
12.    }
13.    public String getPassword() {
14.        return upwd;
15.    }
16.    public void setPassword(String upwd) {
17.        this.upwd = upwd;
18.    }
19.
20.    public String checkLogin() {
21.        if(uname.equals("durga") && upwd.equals("durga")) {
22.            return "success";
23.        }else {
24.            return "failure";
25.        }
26.    }
27.}
```

4.Faces Configuration File:

The main intention of Faces Configuration File in JSF is to provide the following configuration details

1. Managed Beans configuration.
2. Navigations
3. Validators
4. Convertors.

In Simple JSF applications, we will provide managed beans configuration and navigations in Faces Configuration File.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

To prepare Faces Configuration file we have to use the following XML tags.

```
1. <faces-config>
2. <managed-bean>
3.   <managed-bean-name> --- </managed-bean-name>
4.   <managed-bean-class> --- </managed-bean-class>
5.   <managed-bean-scope> --- </managed-bean-scope>
6. </managed-bean>
7. <navigation-rule>
8.   <from-view-id>---</from-view-id>
9.   <navigation-case>
10.     <from-outcome>---</from-outcome>
11.     <to-view-id>---</to-view-id>
12.   </navigation-case>
13.   <navigation-case>
14.     <from-outcome>---</from-outcome>
15.     <to-view-id>---</to-view-id>
16.   </navigation-case>
17. </navigation-rule>
18. </faces-config>
```

- ✓ Where <faces-config> is root tag, it will include JSF configuration details.
- ✓ Where <managed-bean> tag is able to single Managed Bean class configuration.
- ✓ Where <managed-bean-name> is able to take bean logical name of the managed bean class.
- ✓ Where <managed-bean-class> will take fully qualified name of the Managed bean class.
- ✓ Where <managed-bean-scope> will take a particular scope to keep managed bean object.
- ✓ Where <navigation-rule> tag will take the total navigation model of a particular bean.
- ✓ Where <from-view-id> tag will take the name and location of the JSP page from which we are getting request.
- ✓ Where <navigation-case> is providing single forward configuration.
- ✓ Where <from-outcome> tag will take the return value from Business method from Managed Bean class.
- ✓ Where <to-view-id> will take the name and location of the target page to which we want to forward request.

EX:

```
1. <faces-config>
2. <managed-bean>
3.   <managed-bean-name>loginBean</managed-bean-name>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. <managed-bean-class>com.durgasoft.beans.LoginBean</managed-bean-class>
5. <managed-bean-scope>session</managed-bean-scope>
6. </managed-bean>
7. <navigation-rule>
8. <from-view-id>/loginform.jsp</from-view-id>
9. <navigation-case>
10. <from-outcome>success</from-outcome>
11. <to-view-id>/success.jsp</to-view-id>
12. </navigation-case>
13. <navigation-case>
14. <from-outcome>failure</from-outcome>
15. <to-view-id>/failure.jsp</to-view-id>
16. </navigation-case>
17.</navigation-rule>
18.</faces-config>
```

5. Deployment Descriptor or web.xml

In general, in web applications, deployment descriptor is web.xml file, it able to provide the following configuration details.

1. Welcome Files configuration.
2. Display Names Configuration.
3. Servlets Configuration.
4. Filters configuration.
5. Listeners Configurations.
6. Initialization parameters configuration.
7. Context Parameters Configurations.

In JSF based web applications, we have to provide FacesServlet configuration with load-on-startup configuration and we must provide url pattern for FacesServlet in either of the following forms.

/faces/*
*.jsf
*.faces

Example:

```

1. <web-app>
2. <servlet>
3. <servlet-name>facesServlet</servlet-name>
4. <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
5. <load-on-startup>1</load-on-startup>
6. </servlet>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

7. <servlet-mapping>
8.   <servlet-name>facesServlet</servlet-name>
9.   <url-pattern>*.dss</url-pattern>
10. </servlet-mapping>
11.</web-app>
```

Example:

index.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//
/W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <title>Insert title here</title>
7. </head>
8. <body>
9. <jsp:forward page="loginform.dss"/>
10.</body>
11.</html>
```

loginform.jsp

```

1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.   pageEncoding="ISO-8859-1"%>
3. <%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
4. <%@ taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
5. <!DOCTYPE html PUBLIC "-//
/W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
6. <f:view >
7. <html>
8. <body>
9. <h2 style="color: red;" align="center">Durga Software Solutions</h2>
10.<h3 style="color: blue" align="center">User Login Page</h3>
11.<h:form>
12.<center>
13.<h:panelGrid columns="2" >
14. <h:outputText value="User Name"/>
15. <h:inputText id="uname" value="#{loginBean.uname}" />
16.
17. <h:outputText value="Password"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
18. <h:inputSecret id="upwd" value="#{loginBean.upwd}" />
19.
20. <h:commandButton value="Login" action="#{loginBean.checkLogin}" />
21. </h:panelGrid>
22. </center>
23. </h:form>
24. </body>
25. </html>
26. </f:view>
```

success.jsp



```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br>
11. <h1 style="color: red;" align="center">Login Success</h1>
12. </body>
13. </html>
```

failure.jsp



```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2. pageEncoding="ISO-8859-1"%>
3. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4. <html>
5. <head>
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
7. <title>Insert title here</title>
8. </head>
9. <body>
10. <br><br>
11. <h1 style="color: red;" align="center">Login Failure</h1>
12. </body>
13. </html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

LoginBean.java

```

1. package com.durgasoft.beans;
2.
3. public class LoginBean {
4.     private String uname;
5.     private String upwd;
6.
7.     public String getUname() {
8.         return uname;
9.     }
10.    public void setUname(String uname) {
11.        this.uname = uname;
12.    }
13.    public String getUpwd() {
14.        return upwd;
15.    }
16.    public void setUpwd(String upwd) {
17.        this.upwd = upwd;
18.    }
19.
20.    public String checkLogin() {
21.        if(uname.equals("durga") && upwd.equals("durga")) {
22.            return "success";
23.        }else {
24.            return "failure";
25.        }
26.    }
27.}
```

faces-config.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2.
3. <faces-config>
4.     xmlns="http://java.sun.com/xml/ns/javaee"
5.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6.     xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-facesconfig_1_2.xsd"
7.     version="1.2">
8.     <managed-bean>
9.         <managed-bean-name>loginBean</managed-bean-name>
10.        <managed-bean-class>com.durgasoft.beans.LoginBean</managed-bean-class>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11. <managed-bean-scope>session</managed-bean-scope>
12.</managed-bean>
13.<navigation-rule>
14. <from-view-id>/loginform.jsp</from-view-id>
15. <navigation-case>
16.   <from-outcome>success</from-outcome>
17.   <to-view-id>/success.jsp</to-view-id>
18. </navigation-case>
19. <navigation-case>
20.   <from-outcome>failure</from-outcome>
21.   <to-view-id>/failure.jsp</to-view-id>
22. </navigation-case>
23.</navigation-rule>
24.</faces-config>
```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
3. <servlet>
4.   <servlet-name>Faces Servlet</servlet-name>
5.   <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
6.   <load-on-startup>1</load-on-startup>
7. </servlet>
8. <servlet-mapping>
9.   <servlet-name>Faces Servlet</servlet-name>
10.  <url-pattern>*.dss</url-pattern>
11. </servlet-mapping>
12.</web-app>
```

If we want to integrate JSF with Spring Framework then we have to use the following steps.

1. Prepare Presentation part as per JSF Conventions.
2. Prepare Spring Beans and spring configuration file as per Spring Conventions
3. Add Dependent Spring Service Bean to Managed Bean of JSF in faces-config.xml

EX:

```

1. <faces-config>
2. -----
3. <managed-bean>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. <managed-bean-name>loginBean</managed-bean-name>
5. <managed-bean-class>com.durgasoft.beans.LoginBean</managed-bean-class>
6. <managed-bean-scope>session</managed-bean-scope>
7. <managed-property>
8.   <property-name>userService</property-name>
9.   <value>#{userService}</value>
10.  </managed-property>
11. </managed-bean>
12. -----
13. </faces-config>
```

Note: The main intention providing Spring Service Bean in faces-config.xml file is to inject Service bean in Jsf Managed Bean inorder to access Business methods.

4. Provide "SpringBeanFacesELResolver" configuration in faces-config.xml

```

1. <faces-config>
2. -----
3. <application>
4.   <el-resolver>
5.     org.springframework.web.jsf.el.SpringBeanFacesELResolver
6.   </el-resolver>
7. </application>
8. </faces-config>
```

Note: SpringBeanFacesResolver first delegates value lookups to the default resolver of the JSF and then to Spring's WebApplicationContext. This allows to inject springbased dependencies into JSF-managed beans.

5. Provide ContextLoaderListener and RequestContextListener in web.xml file

EX:

web.xml

```

1. <web-app>
2. -----
3. <listener>
4.   <listener-class>
5.     org.springframework.web.context.ContextLoaderListener
6.   </listener-class>
7. </listener>
8.
9. <listener>
10.  <listener-class>
11.    org.springframework.web.context.request.RequestContextListener
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12.    </listener-class>
13.    </listener>
14.    -----
15.    </web-app>
```

Note: These Listeners are used to load the complete Spring context at the time of Startup the Server.

Example:

index.jsp

```

1.  <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.  pageEncoding="ISO-8859-1"%>
3.  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
4.  <html>
5.  <head>
6.  <title>Insert title here</title>
7.  </head>
8.  <body>
9.  <jsp:forward page="loginform.dss"/>
10. </body>
11. </html>
```

loginform.jsp

```

1.  <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2.  pageEncoding="ISO-8859-1"%>
3.  <%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
4.  <%@ taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
5.  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
6.  <f:view >
7.  <html>
8.  <body>
9.  <h2 style="color: red;" align="center">Durga Software Solutions</h2>
10. <h3 style="color: blue" align="center">User Login Page</h3>
11. <h:form>
12. <center>
13. <h:panelGrid columns="2" >
14.   <h:outputText value="User Name"/>
15.   <h:inputText id="uname" value="#{loginBean.uname}">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
16.  
17. <h:outputText value="Password"/>  
18. <h:inputSecret id="upwd" value="#{loginBean.upwd}"/>  
19.  
20. <h:commandButton value="Login" action="#{loginBean.login}"/>  
21.</h:panelGrid>  
22.</center>  
23.</h:form>  
24.</body>  
25.</html>  
26.</f:view>
```

success.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
2. pageEncoding="ISO-8859-1"%>  
3. <!DOCTYPE html PUBLIC "-  
 /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
4. <html>  
5. <head>  
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">  
7. <title>Insert title here</title>  
8. </head>  
9. <body>  
10. <br><br>  
11. <h1 style="color: red;" align="center">Login Success</h1>  
12. </body>  
13. </html>
```

failure.jsp

```
1. <%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
2. pageEncoding="ISO-8859-1"%>  
3. <!DOCTYPE html PUBLIC "-  
 /W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
4. <html>  
5. <head>  
6. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">  
7. <title>Insert title here</title>  
8. </head>  
9. <body>  
10. <br><br>  
11. <h1 style="color: red;" align="center">Login Failure</h1>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
12.</body>
13.</html>
```

LoginBean.java

```
1. package com.durgasoft.beans;
2.
3. import com.durgasoft.service.UserService;
4.
5. public class LoginBean {
6.     private String uname;
7.     private String upwd;
8.     private UserService userService;
9.
10.    public String getUsername() {
11.        return uname;
12.    }
13.    public void setUsername(String uname) {
14.        this.uname = uname;
15.    }
16.    public String getPassword() {
17.        return upwd;
18.    }
19.    public void setPassword(String upwd) {
20.        this.upwd = upwd;
21.    }
22.
23.    public void setUserService(UserService userService) {
24.        this.userService = userService;
25.    }
26.    public UserService getUserService() {
27.        return userService;
28.    }
29.
30.    public String login() {
31.        return userService.checkLogin(uname, upwd);
32.    }
33.}
```

UserService.java

```
1. package com.durgasoft.service;
2.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. public class UserService {
4.     String status = "";
5.     public String checkLogin(String uname, String upwd) {
6.         if(uname.equals("durga") && upwd.equals("durga")) {
7.             status = "success";
8.         }else {
9.             status = "failure";
10.        }
11.        return status;
12.    }
13.}

```

applicationContext.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.         xsi:schemaLocation="http://www.springframework.org/schema/beans
5.                             http://www.springframework.org/schema/beans/spring-beans.xsd">
6.
7.     <bean id="userService" class="com.durgasoft.service.UserService"/>
8. </beans>

```

faces-config.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2.
3. <faces-config
4.     xmlns="http://java.sun.com/xml/ns/javaee"
5.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6.     xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-facesconfig_1_2.xsd"
7.     version="1.2">
8. <application>
9.     <el-resolver>
10.        org.springframework.web.jsf.el.SpringBeanFacesELResolver
11.     </el-resolver>
12. </application>
13. <managed-bean>
14.     <managed-bean-name>loginBean</managed-bean-name>
15.     <managed-bean-class>com.durgasoft.beans.LoginBean</managed-bean-class>
16.     <managed-bean-scope>session</managed-bean-scope>
17.     <managed-property>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

18.    <property-name>userService</property-name>
19.    <value>#{userService}</value>
20.    </managed-property>
21.  </managed-bean>
22.  <navigation-rule>
23.    <from-view-id>/loginform.jsp</from-view-id>
24.    <navigation-case>
25.      <from-outcome>success</from-outcome>
26.      <to-view-id>/success.jsp</to-view-id>
27.    </navigation-case>
28.    <navigation-case>
29.      <from-outcome>failure</from-outcome>
30.      <to-view-id>/failure.jsp</to-view-id>
31.    </navigation-case>
32.  </navigation-rule>
33. </faces-config>
```

web.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.
   org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
   app_4_0.xsd" id="WebApp_ID" version="4.0">
3.    <display-name>jsf_spring_app</display-name>
4.
5.    <servlet>
6.      <servlet-name>Faces Servlet</servlet-name>
7.      <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
8.      <load-on-startup>1</load-on-startup>
9.    </servlet>
10.   <servlet-mapping>
11.     <servlet-name>Faces Servlet</servlet-name>
12.     <url-pattern>*.dss</url-pattern>
13.   </servlet-mapping>
14.   <context-param>
15.     <param-name>javax.faces.PROJECT_STAGE</param-name>
16.     <param-value>Development</param-value>
17.   </context-param>
18.
19.   <!-- Add Support for Spring -->
20.   <listener>
21.     <listener-class>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
22.    org.springframework.web.context.ContextLoaderListener
23.    </listener-class>
24.    </listener>
25.
26.    <listener>
27.      <listener-class>
28.        org.springframework.web.context.request.RequestContextListener
29.      </listener-class>
30.    </listener>
31.  </web-app>
```

To run above web applications we have to use the following jars

1. Jsfjars
2. Springjars
3. Commons-Logging-version.jar

SF Jars:

- + jsf1.2\jsf-api.jar
- + jsf-impl.jar
- + taglibs-standard-impl-1.2.5.jar
- + taglibs-standard-spec-1.2.5.jar

Spring JARs:

- + spring-aop-4.3.9.RELEASE.jar
- + spring-beans-4.3.9.RELEASE.jar
- + spring-context-4.3.9.RELEASE.jar
- + spring-context-support-4.3.9.RELEASE.jar
- + spring-core-4.3.9.RELEASE.jar
- + spring-expression-4.3.9.RELEASE.jar
- + commons-logiin-1.2.jar

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28 **US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

MAVEN

COURSE MATERIAL

BY
NAGOOR BABU

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Index

1. Introduction.....	Page 700
2. Project Object Model.....	Page 701
3. Archetypes in MAVEN.....	Page 713
4. Steps to prepare MAVEN Project in Eclipse.....	Page 716
5. Web Application in Maven.....	Page 728
6. Spring Core Module Application in MAVEN.....	Page 736

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

MAVEN

Introduction

- Maven is a "Yiddish"[German language] word meaning "Accumulator Of Knowledge".
- Maven was originally designed to simplify building processes in Jakarta Turbine project. There were several projects and each project contained slightly different ANT build files. JARs were checked into CVS.
- Apache group then developed Maven which can build multiple projects together, publish projects information, deploy projects, share JARs across several projects and help in collaboration of teams
- MAVEN is a "Project Management Framework", it is much more than a simple Build tool, its declarative and standard approach simplifies many aspects of the Project Lifecycle.

The main Objective of MAVEN is

- A comprehensive model for projects, which is reusable, maintainable, and easier to comprehend [Understand].
- Plugins or tools that interact with this declarative model.

MAVEN follows "Convention over Configuration" Principle, which means that developers are not required to create build process themselves, Developers do not have to mention each and every configuration detail. Maven provides sensible default behavior for projects.

MAVEN does the following activities of the project lifecycle automatically.

- Provides default Project Structer
- Download Required Dependencies [Jars files]
- Compiles Source code
- Packaging projects as .jar, .war, .ear,....
- Starts Server
- Deploying Projects into Servers.
- Perform Unit Testing
- Preparing Test Reports.
- Preparing Documentations
- Undeploy applications from Servers
- Stops Server.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

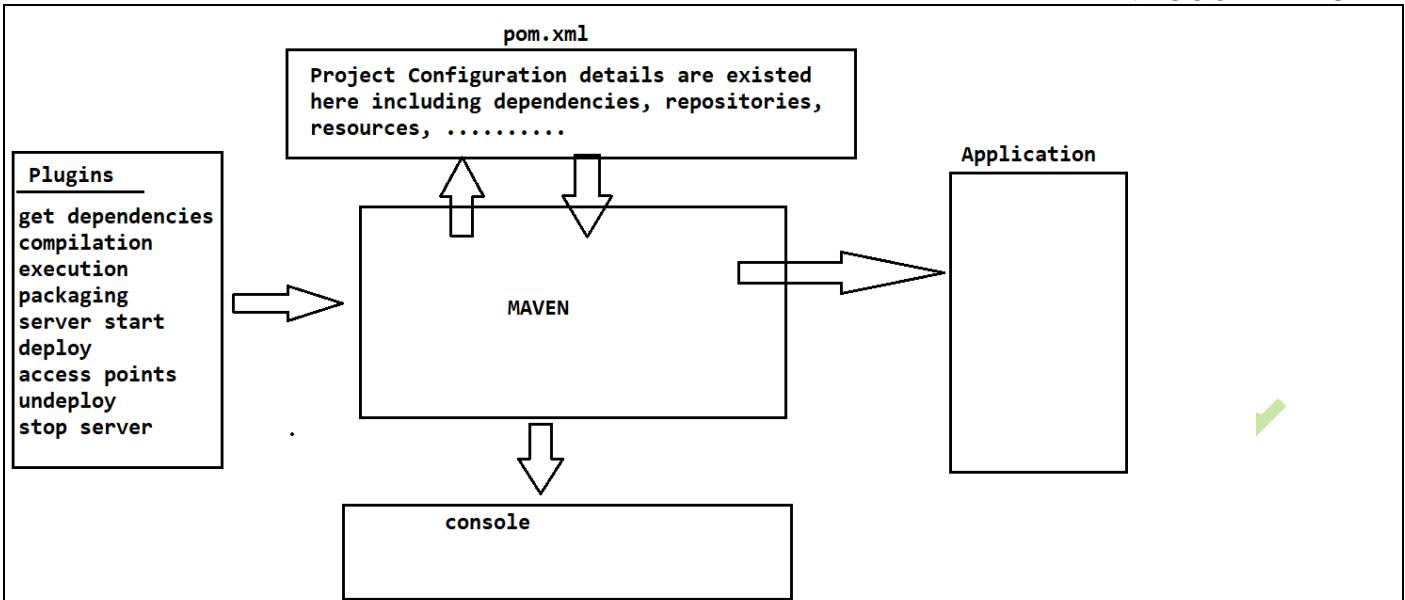
Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU



Project Object Model [pom.xml file]

- i. POM Stands for Project Object Model.
- ii. POM is the fundamental unit in Maven.
- iii. POM is an XML file that contains information about the project and configuration details used by Maven to build the project.
- iv. POM contains default values for projects like build directory, which is target; the source directory, which is src/main/java; the test source directory, which is src/test/java; and so on.
- v. In MAVEN1 , name of pom file is "project.xml", in MAVEN2 it was renamed to pom.xml.
- vi. When we execute MAVEN project then MAVEN will look for the project configurations in pom.xml file and gets the needed things and executes the project.

In Building MAVEN Projects, pom.xml file contains the following configurations.

1. Project Description
2. Repository
3. Dependency Management
4. Project Inheritance
5. Build Configuration
6. Build Profiles

1) Project Description:

In pom file, initial we will identify "Projection Description", it contains Project name, version number, packaged type,.....

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

To specify the above details, we need to use the following XML tags.

```
1. <project ..... >
2. <!-- Project Description -->
3. <modelVersion> --- </modelVersion>
4. <groupid> --- </groupid>
5. <artifactid> -- </artifactid>
6. <version> -- </version>
7. ----
8. ----
9. </project>
```

- Where "<project>" tag is root tag in pom.xml file
- Where "<modelVersion>" tag declared which version of the MAVEN we are using.<modelVersion> tag will take 4.0.0 to support for MAVEN2.x/3.x versions.
- Where "<groupid>" tag will take an unique ID for an organization, or a project.Normally we use a group ID similar to the root Java package name of the project.
- Where "<artifactid>" tag will take name of the project.The artifact ID is used as name for a sub directory under the group ID directory in the Maven repository and as part of the name of the JAR file produced when building the project.The build result, a JAR, WAR or EAR file, is called an artifact in Maven.
- Where "<versin>" tag will take Project version number.
- Where "<packaging>" tag will take different packaging formats inorder to delivery the project like jar, war, ear,...

EX:

```
1. <project ..... >
2. <!-- Project Description -->
3. <modelVersion>4.0.0</modelVersion>
4. <groupid>com.durgasoft.banking</groupid>
5. <artifactid>icici.accounts</artifactid>
6. <version>1.0</version>
7. <name>Account Application</name>
8. <description> Application for Accounts module in icici Bank project </description>
9. ----
10. </project>
```

2) Repository:

If we use Dependencies in MAVEN Project then MAVEN will search for the dependent JARs in Repositories.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

MAVEN will use three types of Repositories in order to get dependencies.

1. Local Repository:

It is a location to manage and supply all dependencies, it will be created by MAVEN when we execute any MAVEN command first time.

In general, MAVEN will create Local Repository at "C:/Users/User_Name/.m2/repository"
EX: C:\Users\LENOVO\.m2\repository

2. Central Repository:

It is a default Repository for MAVEN, it is located at "http://repo1.maven.org/maven2".
IN MAVEN applications, we will use some other repositories are also explicitly like.

1. <http://repository.jboss.org/nexus/content/groups/public>
2. <http://mvnrepository.com>

In MAVEN applications, if we want to use the above explicit repositories then we have to configure them in pom file by using the following xml tags.

```
1. <repositories>
2. <repository>
3. <id>jboss</id>
4. <name>jboss repo</name>
5. <url>http://repository.jboss.org/nexus/content/groups/public/</url>
6. </repository>
7. </repositories>
```

3. Remote Repository:

In some Situations, Maven does not find the dependencies in Local Repository and in central repository, in this context, MAVEN stops the build process and generates some Exceptions. To overcome this problems, Maven has provided a new Features like "Remote Repository".

Remote Repository is a developer's own custom repository containing required libraries or other project jars.

To configure Remote Repository, we have to use the following XML tags in pom.xml file.

```
1. <repositories>
2. <repository>
3. <id>durgasoftware.lib</id>
4. <url>http://library.durgasoftware.com/maven2/lib</url>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftwarelinetraining@gmail.com

WEBSITE: www.durgasoftwareonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

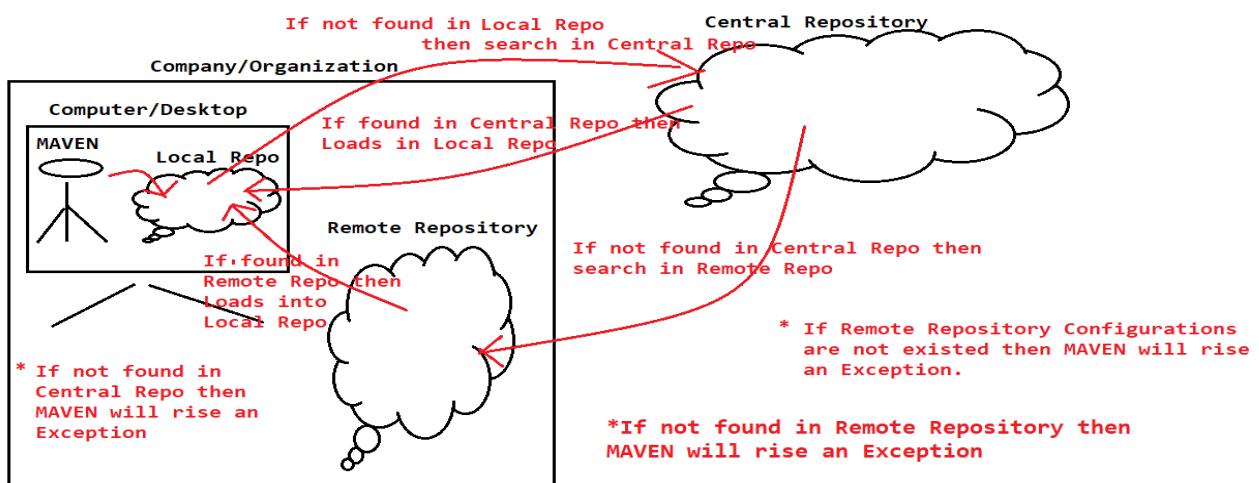


BY NAGOOR BABU

5. </repository>
6. </repositories>

When we run MAVEN project then MAVEN will search for the dependencies in the following order.

- 1) First, MAVEN will search for the dependencies in local repository, if the required dependencies are available at Local Repository the MAVEN will use them in application. If the dependencies are not available at Local Repository then MAVEN search for them at Central Repository.
- 2) If the required Dependencies are existed in central repository then MAVEN will load them into Local Repository and MAVEN will use them in the applications. If the required dependencies are not existed in Central Repository then MAVEN will search for them in Remote Repositories as per configuration.
- 3) If Remote Repository is not configured then MAVEN will stop the application execution and generated some Exceptions.
- 4) If Remote Repository is configured then MAVEN will search for the required dependencies in Remote Repository, if they are identified then MAVEN will load them into Local Repository for futur reference. If the dependencies are not existed at Remote Repositories then MAVEN will stop the execution and generate some Exceptions.



3) Dependency Management:

In Applications, Dependencies are the libraries[Collection of JARs] which are required to compile, test and run our applications.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In General, in application development, we will download the required libraries from internet and we will store them in application directory strucuter.

The main Advantage of MAVEN in applications development is that not to store any Dependent JAR files in Project Directory Strucuter by downloading them explicitly, MAVEN has given flexibility to the developers like to specify dependent JAR files names in pom file, where MAVEN will search for them in the repositories and MAVEN will load them into the project directory strucuter automatically.

If we need any Library in MAVEN based applications then we have to declare them in pom file like below.

```
1. <dependencies>
2.   <dependency>
3.     <groupId>org.hibernate</groupId>
4.     <artifactId>hibernate-core</artifactId>
5.     <version>3.5.6-Final</version>
6.     <scope>provided</scope>
7.   </dependency>
8. </dependencies>
```

If we provide the dependency like above then MAVEN will search for the hibernate library with the name like

<http://repo1.maven.org/maven2/org/hibernate/hibernate-core/3.5.6-Final/>

MAVEN is following "Transitive Dependencies Mechanism", that is, if our dependencies are required any other libraries then MAVEN will get them automatically without loading them explicitly by the developers.

Dependency Scopes:

In Applications, some dependencies are required to all phases of the project lifecycle like compile, test, run,... and some other required only some of phases of the project lifecycle.

In order to limit the dependencies for the lifecycle phases we will use Dependency Scopes.

There are 6 scopes available in MAVEN

1. Compile
2. Provided
3. Runtime
4. Test
5. System

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

6. Import

1. Compile:

It is the default scope in MAVEN . This scope will make the dependencies to avail all phases like compile, test, run,....

EX:

```
<dependency>
```

1. <groupId>org.hibernate</groupId>
2. <artifactId>hibernate-core</artifactId>
3. <version>3.5.6-Final</version>
4. <scope>compile</scope>
5. </dependency>

Note: In general, hibernate-core library is required for all phases of the application.

2. Provided:

This scope will make the dependency libraries to avail upto compilation and upto testing, not for runtime, because, at runtime, JDKs or Containers will provide the required dependencies at runtime.

EX:

In web applications, Servlet API is required explicitly to compile and test the project, but, Servlet API is provided by the container at runtime automatically, so that, they are not required to be exported at runtime.

1. <dependency>
2. <groupId>javax.servlet</groupId>
3. <artifactId> servlet-api</artifactId>
4. <version>3.0.1</version>
5. <scope>provided</scope>
6. </dependency>

3. Runtime:

This scope indicates that the dependency is not required for compilation, but is for execution. It is in the runtime and test class paths, but not the compile class path.

EX:

1. <dependency>

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. <groupId>com.thoughtworks.xstream</groupId>
3. <artifactId>xstream</artifactId>
4. <version>1.4.4</version>
5. <scope>runtime</scope>
6. </dependency>
```

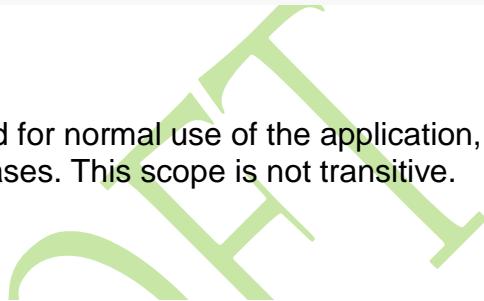
4. Test:

This scope indicates that the dependency is not required for normal use of the application, and is only available for the test compilation and execution phases. This scope is not transitive.

EX:

```

1. <dependency>
2.   <groupId>junit</groupId>
3.   <artifactId>junit</artifactId>
4.   <version>4.12</version>
5.   <scope>test</scope>
6. </dependency>
```

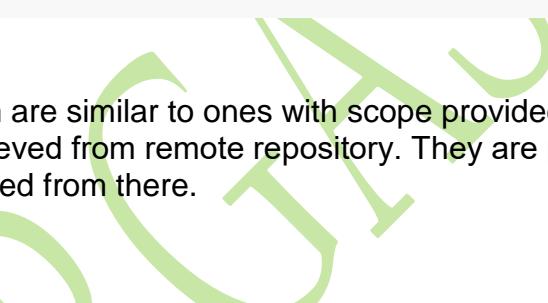
**5. System:**

Dependencies with system are similar to ones with scope provided. The only difference is system dependencies are not retrieved from remote repository. They are present under project's subdirectory and are referred from there.

EX:

```

1. <dependency>
2.   <groupId>Explicit_Dependency</groupId>
3.   <artifactId>Explicit_Dependency</artifactId>
4.   <scope>system</scope>
5.   <version>1.0</version>
6.   <systemPath>apps\app.war\WEB-INF\lib\Explicit_Dependency.jar</systemPath>
7. </dependency>
```

**6. Import:**

It is available in Maven 2.0.9 or later.

Import scope is only supported on a dependency of type pom in the dependencyManagement section. It indicates the dependency to be replaced with the effective list of dependencies in the specified POM's dependencyManagement section.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:

```
1. <dependencyManagement>
2.   <dependencies>
3.     <dependency>
4.       <groupId>other.pom.group.id</groupId>
5.       <artifactId>other-pom-artifact-id</artifactId>
6.       <version>SNAPSHOT</version>
7.       <scope>import</scope>
8.       <type>pom</type>
9.     </dependency>
10.    </dependencies>
11.  </dependencyManagement>
```

4) Project Inheritance:

In MAVEN based applications, it is possible to inherit configurations from one pom file to another pom file inorder to avoid configurations redundancy.

To declare parent pom , we have to use "pom" as value to <packaging> tag in parent pom file.

EX:

```
1. <project ... >
2.   <modelVersion>4.0.0</modelVersion>
3.   <groupId>com.durgasoft</groupId>
4.   <artifactId>my-parent</artifactId>
5.   <version>0.0.1-SNAPSHOT</version>
6.   <packaging>pom</packaging>
7.   ...
8. </project>
```

If we want to inherit parent pom configuration details into a particular chaild pom then we have to configure parent pom in chaild pom.

EX:

```
1. <project ..... >
2. -----
3. <parent>
4.   <groupId>com.durgasoft</groupId>
5.   <artifactId>my-parent</artifactId>
6.   <version>0.0.1-SNAPSHOT</version>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7. </parent>
8. -----
9. <project>
```

Note: In JAVA, java.lang.Object class is common and default super class for every java class inorder to provide 11 common and default methods to each and every java class, similarly , there is a common and default super pom file is existed in maven inorder to provide all common configurations and settings to the chaild pom file.

In general, parent pom contains the following configurations

1. Common data – Developers' names, SCM address, distribution management etc.
2. Constants – Such as version numbers
3. Common dependencies – Common to all child. It has same effect as writing them several times in individual pom files.
4. Properties – For example plugins, declarations, executions and IDs.
5. Configurations
6. Resources

Note: By default, Maven looks for the parent POM first at project's root, then the local repository, and lastly in the remote repository. If parent POM file is not located in any other place, then you can use code tag. This relative path shall be relative to project root.

EX:

```
1. <parent>
2.   <groupId>com.durgasoft</groupId>
3.   <artifactId> MavenExamples </artifactId>
4.   <version>0.0.1-SNAPSHOT </version>
5.   <relativePath>.. /baseapp/pom.xml </relativePath>
6. </parent>
```

Note: If we want to get super pom from MAVEN then use the following command on command prompt from the project root location which contains project specific pom file.

mvn help:effective-pom

Note: Where effective-pm is super pom configurations and project configurations.

5) Build Configuration:

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In MAVEN , Build Configuration is mainly for plugin configurations, resources configurations,.....which are required in MAVEN Project.

MAVEN is simply the collection of plugins, where plugins are used to perform the actions like creating jar files, creating war files, compile Source code, executing unit test code, create project documentation,

MAVEN is having "Plugin Execution Framework" at its heart inorder to execute all plugins.

In MAVEN , there are two types of Plugins.

1. Build Plugins
2. Reporting Plugins

1. Build Plugins: These plugins are executed during the build and they should be configured in the <build/> element from the POM.

EX

1.Clean : It is used when you want to remove files generated at build-time in a project's directory.

2.Compiler: Compiles Java source code.

3.Deploy: It can be used to store artifacts in remote repository while deploying the applications in order to share to other projects .

4.Install: It can be used to install artifacts into local repository.

5.Resources: It will include all the project resources in output directory while creating JAR files.

6.Ear: create ear file from the current project.

7.jar: creates jar file from the current project.

8.war: creates war file from the current project.

9.rar: creates rar file from the current project.

2. Reporting plugins: These plugins are executed during the site generation and they should be configured in the <reporting/> element from the POM.

EX:

1.changelog: Generate a list of recent changes from your SCM[Software Configuration Management].

2.changes: Generate a report from an issue tracker or a change document.

3.javadoc: Generate Javadoc for the project.

4.project-info-reports: Generate standard project reports.

5.surfice-report: Generate a report based on the results of unit tests.

IN general, we will use MAVEN compiler plugin inorder to perform Compilation, for this we have to use the following xml tags in pom.xml file.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
1. <project ---->
2.   <build>
3.     <plugins>
4.       <plugin>
5.         <groupId>org.apache.maven.plugins</groupId>
6.         <artifactId>maven-compiler-plugin</artifactId>
7.         <configuration>
8.           <source>1.8</source>
9.           <target>1.8</target>
10.          </configuration>
11.        </plugin>
12.      </plugins>
13.    </build>
14. </project>
```

By default, all files placed in "src\main\config" are packaged into the generated project artifact and any file which we placed in "src\test\resources" are available in project classpath during unit tests.

If we want to provide our own customized resources location in project then we have to configure them in pom.xml file under <build> tag like below.

```
1. <build>
2.   ---
3.   <resources>
4.     <resource>
5.       <directory>src/main/config</directory>
6.     </resource>
7.     <resource>
8.       <directory>src/main/resources</directory>
9.     </resource>
10.   ---
11.   <resources>
12.   ---
13. </build>
```

6) Build Profiles:

IN general, profiles are used to customize the build lifecycle for different environments like development, testing, production,.....

Example:

```
1. <profiles>
```

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2. <profile>
3.   <id>development</id>
4.   <activation>
5.     <activeByDefault>true</activeByDefault>
6.   </activation>
7.   <properties>
8.     <jdbc.connection.url>jdbc:oracle:thin:@localhost:1521:xe</jdbc.connection.url>
9.   </properties>
10.  </profile>
11.  <profile>
12.    <id>test</id>
13.    <properties>
14.      <jdbc.connection.url>jdbc:mysql://localhost:3306/durgadb</jdbc.connection.url>
15.    </properties>
16.  </profile>
17. </profiles>
```

Where each and every profile has its own id, it can be used to access the respective environment or profile.

In src/main/resources/db.properties

jdbc.connection.url = \${jdbc.connection.url}

If we provide the above setups like above then at compilation time, the respective jdbc URL will be injected to the "jdbc.connection.url" property depending on the target environment.

Use the following command on command prompt inorder to compile the project.

C:/apps>mvn compile

Here "jdbc.connection.profile" property will take "jdbc:oracle:thin:@localhost:1521:xe" value.

C:/apps>mvn compile -Ptest

Here "jdbc.connection.profile" property will take "jdbc:mysql://localhost:3306/durgadb" value.

jdbc.connection.url = \${jdbc.connection.url}

If we provide the above setups like above then at compilation time, the respective jdbc URL will be injected to the "jdbc.connection.url" property depending on the target environment.

Use the following command on command prompt inorder to compile the project.

C:/apps>mvn compile

Here "jdbc.connection.profile" property will take "jdbc:oracle:thin:@localhost:1521:xe" value.

C:/apps>mvn compile -Ptest

Here "jdbc.connection.profile" property will take "jdbc:mysql://localhost:3306/durgadb" value.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Archetypes in MAVEN

MAVEN is providing Standard directory structures to prepare projects depending on the ARCHETYPE selection.

To create sample MAVEN standalone application directory structure we will use the following command on command prompt .

```
C:\mvnapps>mvn archetype:create -DgroupId=com.durgasoft -DartifactId=sampleapp  
-Dpackagename=com.durgasoft.bankapp
```

Steps to Create First Project in MAVEN In Standalone:

- Install MAVEN Software.
- Create MAVEN Project
- Write Java Code
- Compile Java Code
- Execute Java Application

1. Install MAVEN Software:

Installation Process:

1. download apache-maven-3.5.4.zip file from internet[<https://maven.apache.org/download.cgi>]
2. Unzip apache-maven-3.5.4.zip file under C drive and we will get apache-maven-3.5.4 folder.
3. Set the following Environment Variables in System.
 - a) JAVA_HOME: C:/Java/jdk1.8.0;
 - b) path: C:/Java/jdk1.8.0/bin;C:/apache-maven-3.5.4/bin;
 - c) M2_HOME: C:/apache-maven-3.5.4
 - d) MAVEN_HOME: C:/apache-maven-3.5.4 [Optional]

4. Test MAVEN Installation:

Open Command prompt and use the following command.

```
C:\apache-maven-3.5.4\bin>mvn --version
```

```
Apache Maven 3.5.4 (1edded0938998edf8bf061f1ceb3cfdeccf443fe; 2018-06-18T00:03:14+05:30)
```

```
Maven home: C:\apache-maven-3.5.4\bin\..
```

```
Java version: 1.8.0, vendor: Oracle Corporation, runtime: C:\Java\jdk1.8.0\jre
```

```
Default locale: en_US, platform encoding: Cp1252
```

```
OS name: "windows 8.1", version: "6.3", arch: "x86", family: "windows"
```

2. Create MAVEN Project:

- a) Create a separate folder for MAVEN applications

CONTACT US:

Mobile: +91- **8885 25 26 27**

+91- **7207 21 24 27/28**

US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- E:/mvn_projects
- b) Use the following command on Command prompt.
E:/mvn_projects>mvn archetype:generate
 - c) Provide archetype Number: 1202[for maven-archetype-quickstart application]
 - d) Provide archetype version number : 7
 - e) Provide "groupId" value: com.durgasoft
 - f) Provide "archetype" value : sampleapp
 - g) Provide "version" number : 1.0 or 1.0-SNAPSHOT
 - h) Provide "package" name : com.durgasoft
 - i) Provide "Y" to confirm all the above details

If we do the above steps then MAVEN will create quick start project with the following directories and files.

E:/mvn_projects**Sampleapp**

```
|--src
|---main
|   |---java
|   |   |---com
|   |   |   |---durgasoft
|   |   |   |   |---App.java
|---test
|   |---java
|   |   |---com
|   |   |   |---durgasoft
|   |   |   |   |---AppTest.java
|---pom.xml
```

3. Write Java Code:

App.java

```
1. package com.durgasoft;
2.
3. /**
4.  * Hello world!
5.  *
6.  */
7. public class App
```

CONTACT US:

Mobile: +91- 8885 25 26 27 +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
8. {
9.     public static void main( String[] args )
10.    {
11.        System.out.println( "Welcome to Durgasoft" );
12.    }
13.}
```



AppTest.java

```
1. package com.durgasoft;
2.
3. import static org.junit.Assert.assertTrue;
4.
5. import org.junit.Test;
6.
7. /**
8. * Unit test for simple App.
9. */
10. public class AppTest
11.{
12. /**
13. * Rigorous Test :-)
14. */
15. @Test
16. public void shouldAnswerWithTrue()
17. {
18.     System.out.println("Welcome To Durgasoft");
19.     assertTrue( true );
20. }
21.}
```


4. Compile Java Code:

To Compile JAVA code, first, goto application folder then use the following command on command prompt.

E:\mvn_projects\sampleapp>mvn compile

If we use the above command then MAVEN will compile JAVA files and generates the .class files by creating "target" folder under "application folder".

5. Execute MAVEN Project:

IN Two ways we are able to execute MAVEN Project.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. Execute Test case directly
2. Execute Main App by creating project.
- 3.

1) Execute Test case directly:

```
E:\mvn_projects\sampleapp>mvn test
```

2) Execute Main App by creating project.

- a) Create jar file for the project:
`E:\mvn_projects\sampleapp>mvn package`
- b) Set classpath environment variable to the generated jar file:
`E:\mvn_projects\sampleapp>set classpath=E:\mvn_projects\sampleapp\target\sampleapp-1.0.jar;`
Execute Main Class:
`E:\mvn_projects\sampleapp>java com.durgasoft.App`
- c) Welcome to Durasoft

**Steps to prepare MAVEN Project in Eclipse**

1. Create Maven Project in Eclipse:
2. Provide the required configurations in pom.xml
3. Write Application Logic in AppTest.java file
4. Run Maven Project:

1. Create Maven Project in Eclipse:

- 1) Open Eclipse IDE
- 2) Right Click on "Project Explorer"
- 3) Select "New"
- 4) Select "Others"
- 5) Search for "Maven".
- 6) Select for "Maven Project".
- 7) Click on "Next" button.
- 8) Click on "Next" button.
- 9) Provide the following details.
 - a) groupId: com.durgasoft
 - b) artifactId: sampleapp
 - c) version: 0.0.1-SNAPSHOT
 - d) package: com.durgasoft
- 10) Click on "Finish" Button.

CONTACT US:**Mobile: +91- 8885 25 26 27****+91- 7207 21 24 27/28****US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

2. Provide the required configurations in pom.xml

1. Open pom.xml file File and provide java8 plugin configuration.

```
1. <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
3.   <modelVersion>4.0.0</modelVersion>
4.
5.   <groupId>com.durgasoft</groupId>
6.   <artifactId>helloapp</artifactId>
7.   <version>0.0.1-SNAPSHOT</version>
8.   <packaging>jar</packaging>
9.
10.  <name>helloapp</name>
11.  <url>http://maven.apache.org</url>
12.
13.  <properties>
14.    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15.  </properties>
16.
17.  <dependencies>
18.    <dependency>
19.      <groupId>junit</groupId>
20.      <artifactId>junit</artifactId>
21.      <version>3.8.1</version>
22.      <scope>test</scope>
23.    </dependency>
24.  </dependencies>
25.  <build>
26.    <plugins>
27.      <plugin>
28.        <groupId>org.apache.maven.plugins</groupId>
29.        <artifactId>maven-compiler-plugin</artifactId>
30.        <version>3.7.0</version>
31.        <configuration>
32.          <source>1.8</source>
33.          <target>1.8</target>
34.        </configuration>
35.      </plugin>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
36. </plugins>
37. </build>
38.</project>
```

2. Save pom.xml file and see "Markers" for errors.
3. Right Click on :Project"
4. Select "Maven" .
5. Select "Update Project".
6. Click on "OK" button.

3. Write Application Logic in AppTest.java file

src/test/java/AppTest.java



```
1. package com.durgasoft;
2. import junit.framework.Test;
3. import junit.framework.TestCase;
4. import junit.framework.TestSuite;
5. public class AppTest extends TestCase
6. {
7.     public AppTest( String testName )
8.     {
9.         super( testName );
10.    }
11.    public static Test suite()
12.    {
13.        return new TestSuite( AppTest.class );
14.    }
15.    public void testApp()
16.    {
17.        System.out.println("Hello Maven!");
18.        assertTrue( true );
19.    }
20.}
```

Save the above program

4. Run Maven Project:

- ✓ Right Click on Project[Sampleapp].

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ Select "Run As".
- ✓ Select "Maven Test" and see the Result in Console.

To prepare JDBC Application with MAVEN then we have to use the following steps with all the above steps.

1. Install ojdbc6.jar file in Local Repository.
2. Provide ojdbc6.jar file dependency in pom.xml

1. Install ojdbc6.jar file in Local Repository.

a. Keep ojdbc6.jar file in our working directory location.
E:/Mvn_Projects/ ojdbc6.jar

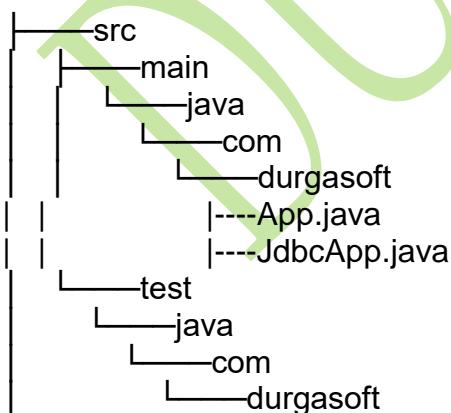
b. use "mvn install:install-file" command on command prompt to install jar file.

```
E:/Mvn_Projects> mvn install:install-file -DgroupId=oracle -DartifactId=oracle-jdbc
-Dpackaging=jar -Dversion=11.2.0-XE -DgeneratePom=true -Dfile=ojdbc6.jar
```

2. Provide ojdbc6.jar file dependency in pom.xml

```
1. <dependencies>
2.  <dependency>
3.    <groupId>oracle</groupId>
4.    <artifactId>oracle-jdbc</artifactId>
5.    <version>11.2.0-XE</version>
6.    <scope>runtime</scope>
7.  </dependency>
8. </dependencies>
```

EX:



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
|      |--AppTest.java  
|---pom.xml
```

pom.xml

```
1. <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
2.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
3.   <modelVersion>4.0.0</modelVersion>  
4.  
5.   <groupId>com.durgasoft</groupId>  
6.   <artifactId>helloapp</artifactId>  
7.   <version>0.0.1-SNAPSHOT</version>  
8.   <packaging>jar</packaging>  
9.  
10.  <name>helloapp</name>  
11.  <url>http://maven.apache.org</url>  
12.  
13.  <properties>  
14.    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
15.  </properties>  
16.  
17.  <dependencies>  
18.    <dependency>  
19.      <groupId>junit</groupId>  
20.      <artifactId>junit</artifactId>  
21.      <version>3.8.1</version>  
22.      <scope>test</scope>  
23.    </dependency>  
24.    <dependency>  
25.      <groupId>oracle</groupId>  
26.      <artifactId>oracle-jdbc</artifactId>  
27.      <version>11.2.0-XE</version>  
28.      <scope>runtime</scope>  
29.    </dependency>  
30.  </dependencies>  
31.  <build>  
32.    <plugins>  
33.      <plugin>  
34.        <groupId>org.apache.maven.plugins</groupId>  
35.        <artifactId>maven-compiler-plugin</artifactId>  
36.        <version>3.7.0</version>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
37.      <configuration>
38.          <source>1.8</source>
39.          <target>1.8</target>
40.      </configuration>
41.      </plugin>
42.  </plugins>
43. </build>
44.</project>
```

JdbcApp.java



```
1. package com.durgasoft;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.ResultSet;
6. import java.sql.Statement;
7.
8. public class JdbcApp {
9.     Connection con;
10.    Statement st;
11.    ResultSet rs;
12.    public JdbcApp() {
13.        try {
14.            Class.forName("oracle.jdbc.OracleDriver");
15.            con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system",
16.                "durga");
17.            st = con.createStatement();
18.        } catch (Exception e) {
19.            e.printStackTrace();
20.        }
21.    }
22.    public void displayEmpDetails() {
23.        try {
24.            rs = st.executeQuery("select * from emp1");
25.            System.out.println("ENO\tENAME\tESAL\tEADDR");
26.            System.out.println("-----");
27.            while(rs.next()) {
28.                System.out.print(rs.getInt(1)+"\t");
29.                System.out.print(rs.getString(2)+"\t");
30.                System.out.print(rs.getFloat(3)+"\t");
31.                System.out.println(rs.getString(4));
32.            }
33.        }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
32.     } catch (Exception e) {  
33.         e.printStackTrace();  
34.     }  
35. }  
36.}
```

AppTest.java



```
1. package com.durgasoft;  
2.  
3. import java.sql.Connection;  
4. import java.sql.DriverManager;  
5. import java.sql.ResultSet;  
6. import java.sql.Statement;  
7.  
8. import junit.framework.Test;  
9. import junit.framework.TestCase;  
10. import junit.framework.TestSuite;  
11.  
12./**  
13. * Unit test for simple App.  
14. */  
15. public class AppTest  
16.     extends TestCase  
17.{  
18.     /**  
19.      * Create the test case  
20.      *  
21.      * @param testName name of the test case  
22.      */  
23.     public AppTest( String testName )  
24.     {  
25.         super( testName );  
26.     }  
27.  
28.     /**  
29.      * @return the suite of tests being tested  
30.      */  
31.     public static Test suite()  
32.     {  
33.         return new TestSuite( AppTest.class );  
34.     }  
35. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

36. /**
37. * Rigorous Test :-)
38. */
39. public void testApp()
40. {
41.     JdbcApp jdbcApp = new JdbcApp();
42.     jdbcApp.displayEmpDetails();
43.     assertTrue( true );
44. }
45.

```

If we want to use MySQL Database in Our project then we have to use the following dependency in pom file.

```

1. <dependency>
2.   <groupId>mysql</groupId>
3.   <artifactId>mysql-connector-java</artifactId>
4.   <version>8.0.11</version>
5. </dependency>

```

In Java Application

Driver Class: com.mysql.cj.jdbc.Driver

Driver URL: jdbc:mysql://localhost:3306/durgadb

DB User Name: root

DB Password : root

Example:

pom.xml

```

1. <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/
   XMLSchema-instance"
2.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/m
   aven-4.0.0.xsd">
3.   <modelVersion>4.0.0</modelVersion>
4.
5.   <groupId>com.durgasoftware</groupId>
6.   <artifactId>helloapp</artifactId>
7.   <version>0.0.1-SNAPSHOT</version>
8.   <packaging>jar</packaging>
9.
10.  <name>helloapp</name>
11.  <url>http://maven.apache.org</url>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftware.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
12.  
13. <properties>  
14.   <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
15. </properties>  
16.  
17. <dependencies>  
18.   <dependency>  
19.     <groupId>junit</groupId>  
20.     <artifactId>junit</artifactId>  
21.     <version>3.8.1</version>  
22.     <scope>test</scope>  
23.   </dependency>  
24.   <!--  
25.   <dependency>  
26.     <groupId>oracle</groupId>  
27.     <artifactId>oracle-jdbc</artifactId>  
28.     <version>11.2.0-XE</version>  
29.     <scope>runtime</scope>  
30.   </dependency>  
31. -->  
32.  
33.   <dependency>  
34.     <groupId>mysql</groupId>  
35.     <artifactId>mysql-connector-java</artifactId>  
36.     <version>8.0.11</version>  
37.   </dependency>  
38.  
39. </dependencies>  
40. <build>  
41.   <plugins>  
42.     <plugin>  
43.       <groupId>org.apache.maven.plugins</groupId>  
44.       <artifactId>maven-compiler-plugin</artifactId>  
45.       <version>3.7.0</version>  
46.       <configuration>  
47.         <source>1.8</source>  
48.         <target>1.8</target>  
49.       </configuration>  
50.     </plugin>  
51.   </plugins>  
52. </build>  
53. </project>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

JdbcApp.java

```
1. package com.durgasoft;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.ResultSet;
6. import java.sql.Statement;
7.
8. public class JdbcApp {
9.     Connection con;
10.    Statement st;
11.    ResultSet rs;
12.    public JdbcApp() {
13.        try {
14.            Class.forName("com.mysql.jdbc.Driver");
15.            con = DriverManager.getConnection("jdbc:mysql://localhost:3300/durgadb", "root", "root");
16.            st = con.createStatement();
17.        } catch (Exception e) {
18.            e.printStackTrace();
19.        }
20.    }
21.    public void displayEmpDetails() {
22.        try {
23.            rs = st.executeQuery("select * from emp1");
24.            System.out.println("ENO\tENAME\tESAL\tEADDR");
25.            System.out.println("-----");
26.            while(rs.next()) {
27.                System.out.print(rs.getInt(1)+"\t");
28.                System.out.print(rs.getString(2)+"\t");
29.                System.out.print(rs.getFloat(3)+"\t");
30.                System.out.println(rs.getString(4));
31.            }
32.        } catch (Exception e) {
33.            e.printStackTrace();
34.        }
35.    }
36.}
```

AppTest.java

```
1. package com.durgasoft;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.  
3. import java.sql.Connection;  
4. import java.sql.DriverManager;  
5. import java.sql.ResultSet;  
6. import java.sql.Statement;  
7.  
8. import junit.framework.Test;  
9. import junit.framework.TestCase;  
10. import junit.framework.TestSuite;  
11.  
12./**  
13. * Unit test for simple App.  
14. */  
15. public class AppTest  
16. extends TestCase  
17.{  
18. /**  
19. * Create the test case  
20. *  
21. * @param testName name of the test case  
22. */  
23. public AppTest( String testName )  
24. {  
25. super( testName );  
26. }  
27.  
28./**  
29. * @return the suite of tests being tested  
30. */  
31. public static Test suite()  
32. {  
33. return new TestSuite( AppTest.class );  
34. }  
35.  
36./**  
37. * Rigorous Test :-)  
38. */  
39. public void testApp()  
40. {  
41. System.out.println("Welcome To Maven");  
42. JdbcApp jdbcApp = new JdbcApp();  
43. jdbcApp.displayEmpDetails();  
44. /*
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
45. Connection con = null;
46. Statement st = null;
47. ResultSet rs = null;
48. try {
49.     Class.forName("oracle.jdbc.OracleDriver");
50.     con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system",
51.         "durga");
52.     st = con.createStatement();
53.     rs = st.executeQuery("select * from emp1");
54.     System.out.println("ENO\tENAME\tESAL\tEADDR");
55.     System.out.println("-----");
56.     while(rs.next()) {
57.         System.out.print(rs.getInt(1)+"\t");
58.         System.out.print(rs.getString(2)+"\t");
59.         System.out.print(rs.getFloat(3)+"\t");
60.         System.out.println(rs.getString(4));
61.     }
62. } catch (Exception e) {
63.     e.printStackTrace();
64. }/*
65. assertTrue( true );
66. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Web Application in Maven

Steps to Prepare Web Application in Maven:

1. Create Web Project in Maven
2. Provide Configurations in pom.xml file
3. Prepare Web Resources.
4. Update Project
5. Run Web Application

1. Create Web Project in Maven:

- i. Click on File in "Eclipse".
- ii. Click on "New".
- iii. Click on "Others".
- iv. Click on "Maven".
- v. Select "Maven Project".
- vi. Click on "Next".
- vii. Click on "Next".
- viii. Select "org.apache.maven.archetypes maven-archetype-webapp 1.0".
- ix. Click on "Next" button.
- x. Provide the following details
 - a. Group Id: com.durgasoft
 - b. Artifact Id: loginapp
 - c. Version: 0.0.1-SNAPSHOT
 - d. package: com.durgasoft
- xi. Click on "Finish" button.

2. Provide Configurations in pom.xml file:

1. Provide Compiler Plugin for JAVA8 version:

```
1. <plugin>
2.   <groupId>org.apache.maven.plugins</groupId>
3.   <artifactId>maven-compiler-plugin</artifactId>
4.   <version>3.7.0</version>
5.   <configuration>
6.     <source>1.8</source>
7.     <target>1.8</target>
8.   </configuration>
9. </plugin>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Provide Tomcat Plugin:

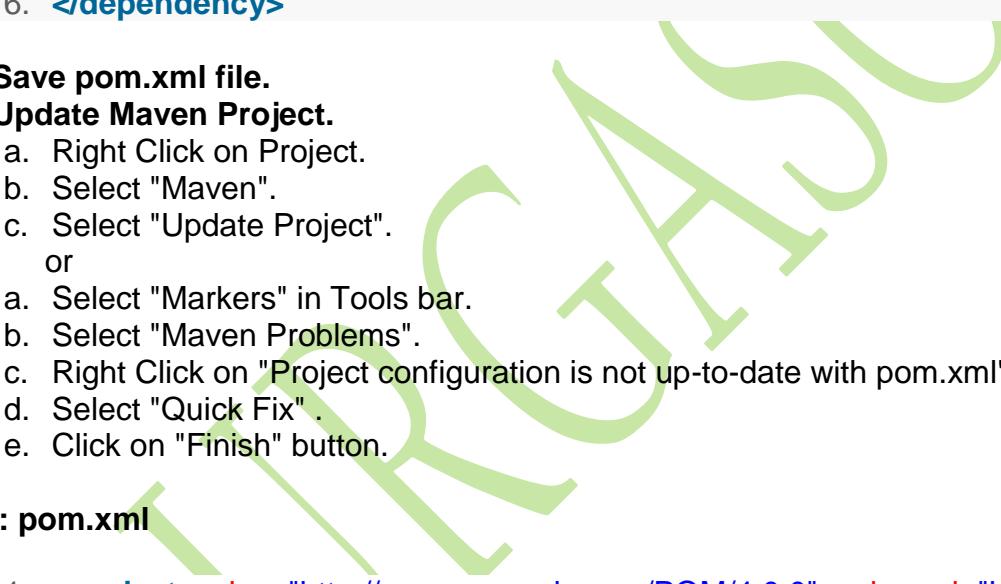
```
1. <plugin>
2.   <groupId>org.apache.tomcat.maven</groupId>
3.   <artifactId>tomcat7-maven-plugin</artifactId>
4.   <version>2.2</version>
5. </plugin>
```

**3 .Provide Servlet API Dependency:**

```
1. <dependency>
2.   <groupId>javax.servlet</groupId>
3.   <artifactId>javax.servlet-api</artifactId>
4.   <version>3.1.0</version>
5.   <scope>provided</scope>
6. </dependency>
```

4. Save pom.xml file.**5. Update Maven Project.**

- Right Click on Project.
- Select "Maven".
- Select "Update Project".
or
- Select "Markers" in Tools bar.
- Select "Maven Problems".
- Right Click on "Project configuration is not up-to-date with pom.xml".
- Select "Quick Fix".
- Click on "Finish" button.

EX: pom.xml

```
1. <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
3.   <modelVersion>4.0.0</modelVersion>
4.   <groupId>com.durgasoftware</groupId>
5.   <artifactId>loginapp</artifactId>
6.   <packaging>war</packaging>
7.   <version>0.0.1-SNAPSHOT</version>
8.   <name>loginapp Maven Webapp</name>
9.   <url>http://maven.apache.org</url>
10.  <dependencies>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftware.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11. <dependency>
12.   <groupId>junit</groupId>
13.   <artifactId>junit</artifactId>
14.   <version>3.8.1</version>
15.   <scope>test</scope>
16. </dependency>
17. <dependency>
18.   <groupId>javax.servlet</groupId>
19.   <artifactId>javax.servlet-api</artifactId>
20.   <version>3.1.0</version>
21.   <scope>provided</scope>
22. </dependency>
23. </dependencies>
24. <build>
25.   <finalName>loginapp</finalName>
26.   <plugins>
27.     <plugin>
28.       <groupId>org.apache.maven.plugins</groupId>
29.       <artifactId>maven-compiler-plugin</artifactId>
30.       <version>3.7.0</version>
31.       <configuration>
32.         <source>1.8</source>
33.         <target>1.8</target>
34.       </configuration>
35.     </plugin>
36.     <plugin>
37.       <groupId>org.apache.tomcat.maven</groupId>
38.       <artifactId>tomcat7-maven-plugin</artifactId>
39.       <version>2.2</version>
40.     </plugin>
41.   </plugins>
42. </build>
43.</project>
```

3. Prepare Web Resources:

a) Prepare html file or jsp file:

1. select "src".
2. select "main".
3. Right Click on "webapps".
4. Select "New".
5. Select "Html File".
6. Provide File Name "loginform.html".

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

7. Click on "Next".
8. Click on "Finish".
9. Provide Html Code in loginform.html.
10. Save Html File.

EX: loginform.html

```
1. <html>
2. <body>
3. <form action=". /hello" method="post">
4.   User Name<input type="text" name="uname"/><br>
5.   Password<input type="password" name="upwd"/><br>
6.   <input type="submit" value="Login"/>
7. </form>
8. </body>
9. </html>
10.
11.success.html
```

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h1>Durga Software Solutions</h1>
9. <h2>User Login Status</h2>
10.<font color="red" size="7">User Login Success</font>
11.<h3>
12.<a href=". /loginform.html">|User Login Form|</a>
13.</h3>
14.</body>
15.</html>
16.
17.failure.html
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h1>Durga Software Solutions</h1>
9. <h2>User Login Status</h2>
10. <font color="red" size="7">User Login Failure</font>
11. <h3>
12. <a href=".//loginform.html">|User Login Form|</a>
13. </h3>
14. </body>
15. </html>
```

b) Prepare Servlets:

- 1) Right Click on Java Resources.
- 2) Select "New".
- 3) Select "Source Folder".
- 4) Provide the following details.
 - a. Project Name : loginapp
 - b. Folder Name: src/main/servlets
 - c. Click on "Finish" Button.
- 5) Right Click on "src/main/servlets" folder.
- 6) Select "New".
- 7) Select "Servlet".
- 8) Provide the following details.
 - a) Package Name: com.durgasoft.servlets
 - b) Class Name : LoginServlet
- 9) Click on "Next" button.
- 10) Edit URL Pattern "/login".
- 11) Click on "Next" button.
- 12) Select "doPost" [bydefault Selected].
- 13) Click on "Finish" button.
- 14) Write Application Logic in Servlets.

EX: LoginServlet.java

```
1. package com.durgasoft.servlets;
2.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. import java.io.IOException;
4.
5. import javax.servlet.RequestDispatcher;
6. import javax.servlet.ServletException;
7. import javax.servlet.http.HttpServlet;
8. import javax.servlet.http.HttpServletRequest;
9. import javax.servlet.http.HttpServletResponse;
10.
11. import com.durgasoft.service.UserService;
12. public class LoginServlet extends HttpServlet {
13.     private static final long serialVersionUID = 1L;
14.     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
15.         response.setContentType("text/html");
16.         String uname = request.getParameter("uname");
17.         String upwd = request.getParameter("upwd");
18.         RequestDispatcher reqDispatcher = null;
19.         UserService userService = new UserService();
20.         String status = userService.checkLogin(uname, upwd);
21.         if(status.equals("success")) {
22.             reqDispatcher = request.getRequestDispatcher("success.html");
23.             reqDispatcher.forward(request, response);
24.         }else {
25.             reqDispatcher = request.getRequestDispatcher("failure.html");
26.             reqDispatcher.forward(request, response);
27.         }
28.
29.     }
30.
31. }
```

web.xml

```

1. <!DOCTYPE web-app PUBLIC
2. "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
3. "http://java.sun.com/dtd/web-app_2_3.dtd" >
4.
5. <web-app>
6.   <display-name>Archetype Created Web Application</display-name>
7.   <servlet>
8.     <servlet-name>LoginServlet</servlet-name>
9.     <display-name>LoginServlet</display-name>
10.    <description></description>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11. <servlet-class>com.durgasoft.servlets.LoginServlet</servlet-class>
12. </servlet>
13. <servlet-mapping>
14. <servlet-name>LoginServlet</servlet-name>
15. <url-pattern>/login</url-pattern>
16. </servlet-mapping>
17. <welcome-file-list>
18. <welcome-file>loginform.html</welcome-file>
19. </welcome-file-list>
20.</web-app>
```

c) Prepare Service Class As per the requirement:

- 1) Right Click on "src/main/servlets"
- 2) Select "New".
- 3) Select "Class"
- 4) Provide the following details.
 - i) Package Name : com.durgasoft.service
 - ii) Class Name : UserService
- 5) Select "Next" button
- 6) Provide Application Logic in UserService class

EX:UserService.java

```
1. package com.durgasoft.service;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.ResultSet;
6. import java.sql.Statement;
7.
8. public class UserService {
9.     Connection con;
10.    Statement st;
11.    ResultSet rs;
12.    String status = "";
13.    public UserService() {
14.        try {
15.            Class.forName("oracle.jdbc.OracleDriver");
16.            con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system",
17.                "durga");
17.            st = con.createStatement();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
18. } catch (Exception e) {
19.     e.printStackTrace();
20. }
21. }
22. public String checkLogin(String uname, String upwd) {
23.     try {
24.         rs = st.executeQuery("select * from reg_Users where uname = '"+uname+"' and up
25.             wd = '"+upwd+"'");
26.             boolean b = rs.next();
27.             if(b == true) {
28.                 status = "success";
29.             }else {
30.                 status = "failure";
31.             }
32.     } catch (Exception e) {
33.         e.printStackTrace();
34.     }
35.     return status;
36. }
```

4. Update Project:

1. Right Click on Project.
2. Select "Maven".
3. Select "Update Project".
4. Select Project and Click on "OK" button.

5. Run Application:

1. Right Click on Project.
2. Select "Run As".
3. Select "Maven Build...".
4. Provide Value to "Goals" : tomcat:run -Dmaven.tomcat.port=9944[Any port number]
5. Click on "Run" button.
6. Copy the URL from Console.
7. Past the URL in Browser Addressbar.
<http://localhost:9944/loginapp/loginform.html>
8. Provide data and click on "Login" button and get response from LoginServlet.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Spring Core Module Application in MAVEN

Steps to Prepare Spring Core Module Application in MAVEN:

1. Create Maven Quick Start Project.
2. Provide Configurations in pom.xml file.
3. Prepare Spring Resources like Beans, Configuration Files, Test and Cases...
4. Run Spring Application

1. Create Maven Quick Start Project.

1. Open Eclipse IDE
2. Right Click on "Project Explorer"
3. Select "New"
4. Select "Others"
5. Search for "Maven".
6. Select for "Maven Project".
7. Click on "Next" button.
8. Click on "Next" button.
9. Provide the following details.
 - a. groupId: com.durgasoft
 - b. artifactId: springapp1
 - c. version: 0.0.1-SNAPSHOT
 - d. package: com.durgasoft
10. Click on "Finish" Button.

2. Provide Configurations in pom.xml file.

1. Open pom.xml file File and provide java8 plugin configuration:

EX:

```
1. <build>
2.   <plugins>
3.     <plugin>
4.       <groupId>org.apache.maven.plugins</groupId>
5.       <artifactId>maven-compiler-plugin</artifactId>
6.       <version>3.7.0</version>
7.       <configuration>
8.         <source>1.8</source>
9.         <target>1.8</target>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10.    </configuration>
11.   </plugin>
12.  </plugins>
13. </build>
```

2. Provide Spring dependencies in pom.xml file



EX:

```

1. <dependencies>
2.   <dependency>
3.     <groupId>org.springframework</groupId>
4.     <artifactId>spring-core</artifactId>
5.     <version>4.3.18.RELEASE</version>
6.   </dependency>
7.
8.   <dependency>
9.     <groupId>org.springframework</groupId>
10.    <artifactId>spring-context</artifactId>
11.    <version>4.3.18.RELEASE</version>
12.  </dependency>
13.
14.  <dependency>
15.    <groupId>org.springframework</groupId>
16.    <artifactId>spring-aop</artifactId>
17.    <version>4.3.18.RELEASE</version>
18.  </dependency>
19. </dependencies>
```

Note: In the above dependencies, just add spring-core and spring-context dependencies in pom.xml file then automatically all other dependencies are also be loaded.

EX: pom.xml

```

1. <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/
   XMLSchema-instance"
2.   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/m
   aven-4.0.0.xsd">
3.   <modelVersion>4.0.0</modelVersion>
4.
5.   <groupId>com.durgasoft</groupId>
6.   <artifactId>springapp1</artifactId>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28


US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7. <version>0.0.1-SNAPSHOT</version>
8. <packaging>jar</packaging>
9.
10. <name>springapp</name>
11. <url>http://maven.apache.org</url>
12.
13. <properties>
14.   <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15. </properties>
16.
17. <dependencies>
18.   <dependency>
19.     <groupId>junit</groupId>
20.     <artifactId>junit</artifactId>
21.     <version>3.8.1</version>
22.     <scope>test</scope>
23.   </dependency>
24.
25.   <dependency>
26.     <groupId>org.springframework</groupId>
27.     <artifactId>spring-core</artifactId>
28.     <version>4.3.18.RELEASE</version>
29.   </dependency>
30.
31.   <dependency>
32.     <groupId>org.springframework</groupId>
33.     <artifactId>spring-context</artifactId>
34.     <version>4.3.18.RELEASE</version>
35.   </dependency>
36.
37.   <dependency>
38.     <groupId>org.springframework</groupId>
39.     <artifactId>spring-aop</artifactId>
40.     <version>4.3.18.RELEASE</version>
41.   </dependency>
42. </dependencies>
43. <build>
44.   <plugins>
45.     <plugin>
46.       <groupId>org.apache.maven.plugins</groupId>
47.       <artifactId>maven-compiler-plugin</artifactId>
48.       <version>3.7.0</version>
49.     <configuration>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

50.      <source>1.8</source>
51.      <target>1.8</target>
52.    </configuration>
53.  </plugin>
54. </plugins>
55. </build>
56. </project>
```

2. Save pom.xml file and see "Markers" for errors.
3. Right Click on :Project"
4. Select "Maven" .
5. Select "Update Project".
6. Click on "OK" button.

3. Prepare Spring Resources like Beans, Configuration Files, Test Cases...

a) Prepare Bean class:

1. Right Click on "src/java/main".
2. Select "New"
3. Select "Class".
4. Provide the following details
Package: com.durgasoft.beans
Class : HelloBean
- 5 .Click on "Finish" button.
6. Provide properties and methods in Bean class.

EX: HelloBean.java

```

1. package com.durgasoft.beans;
2.
3. public class HelloBean {
4.     public String sayHello() {
5.         return "Hello User!";
6.     }
7. }
```

b) Prepare Spring Configuration File:

1. Right Click on "src\java\main".
2. Select "Package".
3. Provide package name "com.durgasoft.resources".
4. Click on "Finish" button.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

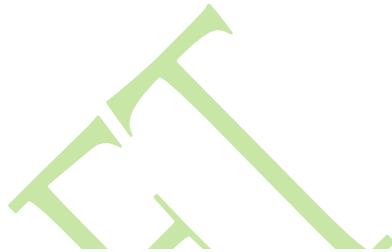
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

5. Right Click on "com.durgasoft.resources" package.
6. Select "New".
7. Select "Other".
8. Search "XML" and select "XML File".
9. Click on "Next" button.
10. Provide file Name: applicationContext.xml
11. Click on "Next" button.
12. Click on "Finish" Button.

**EX: applicationContext.xml**

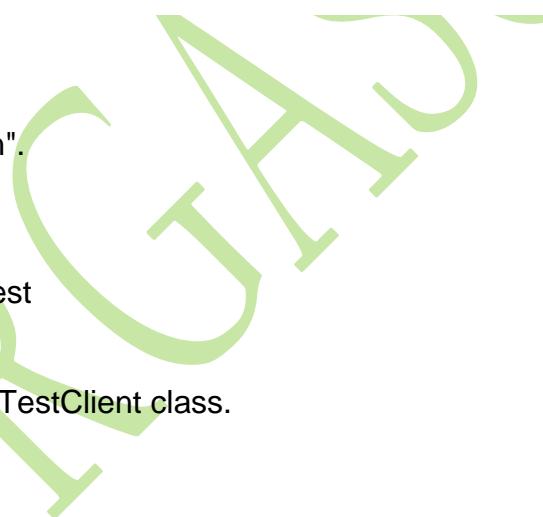
```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <beans xmlns="http://www.springframework.org/schema/beans"
3.   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.   xsi:schemaLocation="http://www.springframework.org/schema/beans
5.     http://www.springframework.org/schema/beans/spring-beans.xsd">
6.   <bean id="helloBean" class="com.durgasoft.beans.HelloBean"/>
7. </beans>
```

c) Prepare Test Application:

1. Right Click on "src/java/main".
2. Select "New"
3. Select "Class".
4. Provide the following details

Package: com.durgasoft.test
 Class : TestClient
5. Click on "Finish" button.
6. Provide application Logic in TestClient class.

**EX:TestClient.java**

```

1. package com.durgasoft.test;
2.
3. import org.springframework.context.ApplicationContext;
4. import org.springframework.context.support.ClassPathXmlApplicationContext;
5.
6. import com.durgasoft.beans.HelloBean;
7.
8. public class TestClient {
9.   public void test() {
10.    ApplicationContext context = new ClassPathXmlApplicationContext("com/durgasoft/re
sources/applicationContext.xml");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11. HelloBean bean = (HelloBean) context.getBean("helloBean");
12. String message = bean.sayHello();
13. System.out.println(message);
14. }
15.

```

d) Access TestClient Application from JUnit test[src\test\main\TestApp.java]:**EX:TestApp.java**

```

1. package com.durgasoft;
2. import com.durgasoft.test.TestClient;
3. import junit.framework.Test;
4. import junit.framework.TestCase;
5. import junit.framework.TestSuite;
6. public class AppTest
7. extends TestCase
8. {
9.     public AppTest( String testName )
10.    {
11.        super( testName );
12.    }
13.
14.    public static Test suite()
15.    {
16.        return new TestSuite( AppTest.class );
17.    }
18.
19.    public void testApp()
20.    {
21.        TestClient client = new TestClient();
22.        client.test();
23.
24.        assertTrue( true );
25.    }
26.

```

**4. Run Spring Application:**

1. Right Click on Project[springapp1].
2. Select "Run As".
3. Select "Maven Test" and see the Result in Console.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGA SOFT

LOG4J

COURSE MATERIAL

BY

NAGOOR BABU

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

SPRING LOG4J MODULE INDEX

1. INTRODUCTION.....	Page 745
2. Log4j.....	Page 748
3. Appenders.....	Page 755
4. Layouts.....	Page 761

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

INTRODUCTION

In Real time applications development, we may get no of problems or bugs or exceptions while executing or testing the applications, To identify the problems and their locations then we have to trace the applications flow of execution.

To trace applications flow of execution we will use "System.out.println(--)" at basic level.

EX:

```
1. class Transaction{  
2.     void deposit(){  
3.         System.out.println("logic for deposit");  
4.     }  
5.     void withdraw(){  
6.         System.out.println("logic for withdraw");  
7.     }  
8.     void transfer(){  
9.         System.out.println("logic for transfer");  
10.    }  
11.}  
12. class Test{  
13.     Transaction tx = new Transaction();  
14.     System.out.println("Before deposit() method call");  
15.     tx.deposit()  
16.     System.out.println("After deposit() method call");  
17.     ----  
18.     System.out.println("Before withdraw() method call");  
19.     tx.withdraw();  
20.     System.out.println("After withdraw method call");  
21.     -----  
22.     System.out.println("Before transfer() method call");  
23.     tx.transfer();  
24.     System.out.println("After transfer() method call");  
25. }
```

If we execute the above application then we are able to get the following output on console or command prompt.

- ✓ Before deposit() method call
- ✓ logic for deposit
- ✓ After deposit() method call
- ✓ Before withdraw() method call
- ✓ logic for withdraw

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ After withdraw() method call
- ✓ Before transfer() method call
- ✓ logic for transfer
- ✓ After transfer() method call

If we use `System.out.println()` method in applications to trace flow of execution then we are able to get the following problems.

1. `System.out.println()` will be used for only console display, not for sending data to any other output systems like file systems[html files, xml files, text files...], databases, network,....
2. `System.out.println()` method contains synchronized block to display data, it is heavy weight, expensive and time consuming.
3. In Applications, it is not suggestible to write too many no of `System.out.println()` methods in applications, because, it will reduce application performance.
4. `System.out.println()` method is usefull in Development environment only, It is not suitable in production environment , because, if we use `System.out.println()` method in server side applications then it will display messages in servers console only, it will not be available to users.
5. `System.out.println()` will display the messages on console or on command prompt, it will not show differences in Error messages, warning messaages, normal information,....
6. `System.out.println()` is suitable for simple standalone applications, not for complex enterprise applications.

To overcome all the problems while tracing applications we have to use Logging.

Logging : It is the process of writing log messages during the execution of a program to a central place. This logging allows you to report and persist error and warning messages as well as info messages so that the messages can later be retrieved and analyzed.

In general, in java applications, Logging is required

1. To understand flow of execution in applications
2. To manage exception messages when Exceptions are occurred in java applications
3. To manage the event - notification messages in file systems....

Logging Framework: It is a set of classes and interfaces or a product to perform Logging in Java applications.

EX:

1. Java provided `java.util.Logging`
2. Log4j
3. LogBack
4. Commons Logging

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

5. ObjectGuy
6. TinyLog

Advantages of Logging Frameworks:

- 1. Problem diagnosis:** In general, in application development, we are able to detect the problems which are occurred in compilation time or in execution, but, some hidden problems are existed inside the applications, which may not come outside directly, but, they will give effect to our application execution, in this situation, if we use good Logging Framework then it is possible to detect the hidden problems quickly and we can fix them easily.
- 2. Quick Debugging:** Once if we diagnose the problem and if we identify the location of the problem by using logging framework then it is very simple to fix that problem and it simplifies Debugging and it able to reduce overall Debugging time.
- 3. Easy Maintenance:** If we provide good Logging Framework to perform logging in our applications then it will provide some description or information about our application which is useful to manage our applications easily.
- 4. History:** In general, all Logging Frameworks are tracing the applications flow of execution and they are able to store tracing details in a file system, this logging details are useful to analyze the problems and to solve the problems.
- 5. Cost and Time Saving:** Logging Frameworks will simplify debugging, easy maintenance , persisting application information,..... , these qualities of Logging Framework saves time and cost of the application.

Drawbacks with Logging Frameworks:

1. Logging Frameworks need to write some extra code in our java applications, it will increase overhead to the application.
2. Logging Frameworks will provide some extra code in our applications, it will increase application execution time at runtime, it will reduce application performance.
3. Logging Framework will provide some extra code in application, so that, it will increase application length.
4. Poor logging strategies will increase confusion to the developers.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



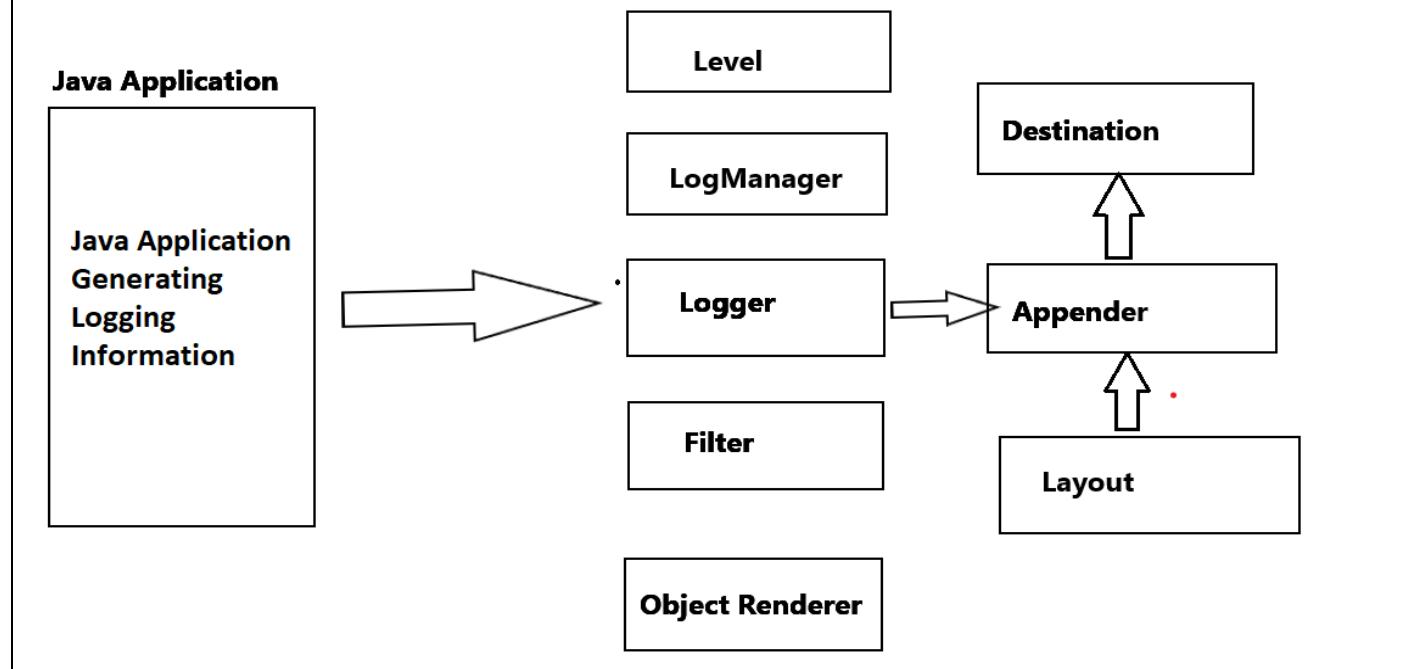
Log4j

Log4J is a reliable, fast and flexible framework written in JAVA to perform logging in Java applications and it is provided by Apache Software Foundations.

Log4j Features:

- + Log4j is able to allow more than one thread at a time without providing data inconsistency, so that, Log4j is thread-safe.
 - + Log4j will not slow down the application execution, it will be optimized to improve application performance.
 - + Log4j is providing environment to send logging messages to more than one output appenders .
 - + Log4j supports internationalization.
 - + Log4j is providing services for both predefined and user defined facilities, it able to provide some customizations also.
 - + Log4j allows to set logging behaviours at runtime through the configuration file.
 - + Log4j is providing very good environment to trace Exceptions from its root.
 - + Log4j uses multiple levels like ALL, TRACE, DEBUG, INFO, WARN, ERROR and FATAL to generate messages.
 - + Log4j allows to change the format of log output by extending Layout class.
 - + Log4j is using appenders to generate log messages.

Log4j Architecture:





BY NAGOOR BABU

Log4j Arch contains mainly the following Objects:

1. Logger
2. Appender
3. Layout
4. Level
5. LogManager
6. Filter
7. ObjectRender

1. Logger:

This Object is responsible to get logging messages from Java applications

2. Appender:

The main intention of Appender object is to get logging messages from Logger object and publishing that logging messages to the preferred destinations. Each and Every Appender Object must have atleast one Destination object inorder to send logging messages.

EX: ConsoleAppender is able to store logging messages on Console.

3. Layout:

The main intention of Layout object is to format logging messages in different styles. Layout Object is used by Appender object just before publishing Logging Messages.

4. Level:

The main intention of Level object is to define granularity and priority of any Logging information. Each and every Logging information must be with a particular Logging Level.

Log4j API has provided the following Levels to the Logging messages.

1. ALL
2. TRACE
3. DEBUG
4. INFO
5. WRAN
6. ERROR
7. FATAL
8. OFF

Log4j is giving priorities for the logging messages in the following order.
ALL >TRACE > DEBUG > INFO > WARN > ERROR > FATAL>OFF

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

6. LogManager

LogManager is the central component , it able to manage Log4j framework, it will read all the initial configuration details from the configuration file and stored the Logger objects in namespace hierarchy with a particular reference name for futer reference. If our application access any Logger object on the basis of the reference name then LogManager will return the existed Logger object otherwise it will create new Logger object and return to the application.

7. Filter:

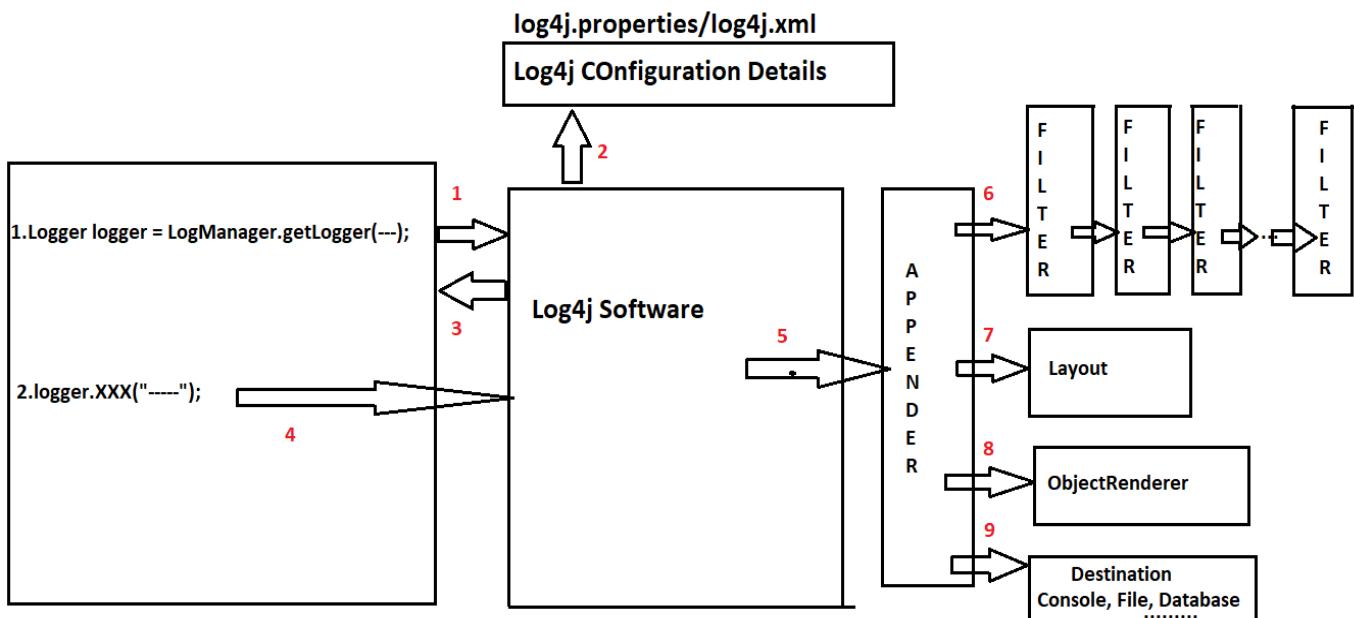
The main intention of Filter object is used to analyze logging information and takes the decision whther the messages must be logged or not.

An Appender object may have no of Filter objects, they must approve our logging message before publishing logging messages in destination.

8.ObjectRender

This Object will be used by Layout object in order to get string representation of the several objects which are passed through Log4j framework.

Log4j Internal Flow:



1. If we create Logger object in Java application then a request will be send to Log4j Software about to create Logger object.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Log4j Software will search for log4j.properties/log4j.xml file in classpath, if it is existed then Log4j Software will load and parse log4j configuration file then Log4j software will create Logger object.
3. After creating Logger object, Log4j Software will return Logger object to Java Application.
4. If we access any logger method from Logger reference then Java application will send logging responsibility to Log4j software.
5. Log4j Software will activate Appender object inorder to perform logging.
6. Appender will recognize all the filters which are associated then Appender will execute all the filters in sequence, where filters will filters the logging messages to persist or display.
7. Appender will activate Layout object , it will apply the layout styles on the logging messages.
8. After Styling Logging messages, Appender will activate Object Renderer to recognize the destination.
9. Sending Logging messages to the respective Destination.

Log4j Configuration File:

Log4j Configuration File includes all Log4j configuration details like rootLogger, Layout configurations, Appenders configuration,,.

In Log4j , we are able to use the following two types of configuration files.

- 1.log4j.properties
- 2.log4j.xml

IN both the configuratin files, we have to use the following configurations at basic java applications.

1. Declare rootLogger and initialize rootLogger
2. Declare appender
3. Declare Layout and its conversionPattern.

EX: log4j.properties:

- ✓ log4j.rootLogger =TRACE, CONSOLE
- ✓ log4j.appender.CONSOLE = org.apache.log4j.ConsoleAppender
- ✓ log4j.appender.CONSOLE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.CONSOLE.layout.conversionPattern=%d{yyyy-mm-dd hh:mm:ss } : This is %m%n
- Where "log4j.rootLogger" property will take Logging LEVEL and appender name,
- where Log4j framework will display all the messages which are same as the specified Log level and lower than the specified Log level.

CONTACT US:

Mobile: +91- **8885 25 26 27**

 +91- **7207 21 24 27/28**



US NUM: **4433326786**

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- Where appender name is a name which is referring a particular Appender.

EX: log4j.rootLogger = TRACE, CONSOLE

Where "log4j.appender.CONSOLE" property will take a particular Appender like ConsoleAppender, FileAppender,...

EX: log4j.appender.CONSOLE = org.apache.log4j.ConsoleAppender

Where "log4j.appender.CONSOLE.layout" property will configure Layout configuration.

EX: log4j.appender.CONSOLE.layout = org.apache.log4j.PatternLayout

Where PatternLayout needs a property like conversionPattern to take a particular pattern to display messages.

EX: "log4j.appender.CONSOLE.layout.conversionPattern = %m%n

Note: Where %m%n in the sense, %m is to display message and %n is to bring cursor to next line.

log4j.xml Configuration:

In XML configuration we have to use the following configuration.

- Appender configuration.
- Layout Configuration
- Root Logger Configuration

To provide XML configuration then we have to use the following xml tags in log4j.xml file.

log4j.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
3. <log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
4.   debug="false">
5.
6.   <appender name="console" class="org.apache.log4j.ConsoleAppender">
7.     <layout class="org.apache.log4j.PatternLayout">
8.       <param name="ConversionPattern" value="%d{DD/MM/YYYY HH:mm:ss} %-
5p %c{1} - %m%n" />
9.     </layout>
10.    </appender>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11.  
12.<root>  
13. <priority value="ERROR" />  
14. <appender-ref ref="console" />  
15.</root>  
16.  
17.</log4j:configuration>
```

- ✓ Where <log4j:configuration> is a root tag, it able to take log4j configuration details.
- ✓ Where <appender> tag is able to configure Appender class with a logical name.
- ✓ Where "name" attribute in <appender> tag will take logical name to the appender.
- ✓ Where "class" attribute in <appender> tag is able to take fully qualified name of the Appender class, that is, "org.apache.log4j.ConsoleAppender"
- ✓ Where <layout> tag can be used to configure Layout class.
- ✓ Where "class" attribute in <layout> tag will take fully qualified name of Layout class, that is, "org.apache.log4j.PatternLayout".
- ✓ Where <param> tag in <layout> tag will take layout parameter .
- ✓ Where "name" attribute will take parameter name, tyhat is , "ConversionPattern".
- ✓ Where "value" attribute will take value of the parameter , that is, %d{DD/MM/YYYY HH:mm:ss} %-5p %c{1} - %m%n
- ✓ Where <root> tag will provide rootLogger configuration.
- ✓ Where <priority> tag tag will take logger level with "value" attribute.
- ✓ Where <appender-ref> tag will take logical name of the appender with "ref" attribute.

Steps to Use Log4j in Java Application:

1. Download log4j-1.2.17.zip from "<https://logging.apache.org/log4j/1.2/download.html>"
2. Create Java project in Eclipse IDE and add log4j1.2.17.jar file in build path
3. Create log4j configuration file[log4j.properties or log4j.xml].
4. Create Java class.
5. Run Java project [Suggestible in Debug mode].

Example:

log4j.properties

- ✓ log4j.rootLogger =FATAL, CONSOLE
- ✓ log4j.appender.CONSOLE = org.apache.log4j.ConsoleAppender
- ✓ log4j.appender.CONSOLE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.CONSOLE.layout.conversionPattern = %d{yyyy-mm-dd hh:mm:ss } : This is %m%n

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Log4jTest.java

```
1. package log4japp1;
2.
3. import org.apache.log4j.LogManager;
4. import org.apache.log4j.Logger;
5.
6. public class Log4jTest {
7.
8.     public static void main(String[] args) {
9.         Logger logger = LogManager.getLogger(Log4jTest.class);
10.        logger.trace("trace Message");
11.        logger.debug("debug Message");
12.        logger.info("info Message");
13.        logger.warn("warn Message");
14.        logger.error("Error Message");
15.        logger.fatal("fatal Message");
16.
17.    }
18.}
```

Note: If we want to use XML configuration then we have to use the following XML file **log4j.xml**

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
3. <log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/"
4.   debug="false">
5.
6. <appender name="console" class="org.apache.log4j.ConsoleAppender">
7.   <layout class="org.apache.log4j.PatternLayout">
8.     <param name="ConversionPattern" value="%d{DD/MM/YYYY HH:mm:ss} %-
5p %c{1} - %m%n" />
9.   </layout>
10. </appender>
11.
12. <root>
13.   <priority value="ERROR" />
14.   <appender-ref ref="console" />
15. </root>
16.
17. </log4j:configuration>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Appenders

Appenders:

Appender is an object, it can be used to get logging messages from Logger object and publishing that logging messages to the preferred destinations. Each and Every Appender Object must have atleast one Destination object inorder to send logging messages.

Log4j has provided the following appenders inorder to publish logging messages.

1. ConsoleAppender
2. FileAppender
3. JDBCAppender
4. JMSAppender
5. SocketAppender
6. SMTPAppender
7. AsyncAppender
8. AppenderSkeleton
9. DailyRollingFileAppender
10. ExternallyRolledFileAppender
11. LF5Appender
12. NTEventLogAppender
13. NullAppender
14. RollingFileAppender
15. SocketHubAppender
16. SyslogAppender
17. TelnetAppender
18. WriterAppender

If we want to manage all logging messages in a File then we have to use FileAppenders.

There are Four types of File Appenders.

1. FileAppender
2. RollingFileAppender
3. DailyRollingFileAppender
4. ExternallyRolledFileAppender

1. FileAppender:

If we want to use FileAppender in Java applications then we have to use the following properties in log4j.properties file.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. log4j.rootLogger
2. log4j.appender.Ref_Name
3. log4j.appender.Ref_Name.File : It will name and location of the file to manage logging messages.
4. log4j.appender.Ref_Name.layout
5. log4j.appender.Ref_Name.layout.ConversionProperty

EX:**log4j.properties**

- ✓ log4j.rootLogger =ALL, FILE
- ✓ log4j.appender.FILE = org.apache.log4j.FileAppender
- ✓ log4j.appender.FILE.File=./logging/logs.log
- ✓ log4j.appender.FILE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.FILE.layout.conversionPattern = %d{yyyy-mm-dd hh:mm:ss } : This is %m%n

Log4jTest.java

```
1. package log4japp1;
2.
3. import org.apache.log4j.LogManager;
4. import org.apache.log4j.Logger;
5.
6. public class Log4jTest {
7.
8.     public static void main(String[] args) {
9.
10.     Logger logger = LogManager.getLogger(Log4jTest.class);
11.     logger.trace("trace Message");
12.     logger.debug("debug Message");
13.     logger.info("info Message");
14.     logger.warn("warn Message");
15.     logger.error("Error Message");
16.     logger.fatal("fatal Message");
17. }
18. }
19. }
```

If we want to use both ConsoleAppender and FileAppender in Java Applications then we have to define two Ref_Names at 'log4j.rootLogger' property.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:**log4j.properties**

log4j.rootLogger =ALL, CONSOLE, FILE

#ConsoleAppender Configuration

- ✓ log4j.appender.CONSOLE = org.apache.log4j.ConsoleAppender
- ✓ log4j.appender.CONSOLE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.CONSOLE.layout.conversionPattern = %d{yyyy-mm-dd hh:mm:ss } %-c %p : This is %m%n

#FileAppender Configuration

- ✓ log4j.appender.FILE = org.apache.log4j.FileAppender
- ✓ log4j.appender.FILE.File=./logging/logs.log
- ✓ log4j.appender.FILE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.FILE.layout.conversionPattern = %d{yyyy-mm-dd hh:mm:ss } %-c %p : This is %m%n

Log4jTest.java

```
1. package log4japp1;
2.
3. import org.apache.log4j.LogManager;
4. import org.apache.log4j.Logger;
5.
6. public class Log4jTest {
7.
8.     public static void main(String[] args) {
9.
10.         Logger logger = LogManager.getLogger(Log4jTest.class);
11.         logger.trace("trace Message");
12.         logger.debug("debug Message");
13.         logger.info("info Message");
14.         logger.warn("warn Message");
15.         logger.error("Error Message");
16.         logger.fatal("fatal Message");
17.
18.     }
19. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. RollingFileAppender

The main intention of RollingFileAppender is to manage logging messages to the other new files when the present log file is getting exceeded as per the file size limit.

If we want to use RollingFileAppender in log4j applications then we have to use "org.apache.log4j.RollingFileAppender" class and we have to use all the FileAppender properties and the following extra properties.

1. log4j.appenders.Ref_Name.MaxFileSize : It will take max File size in the form of KB
2. log4j.appenders.Ref_Name.MaxBackupIndex= It able to create maximum backup files.

EX:**log4j.properties**

```
log4j.rootLogger =ALL, FILE
```

#FileAppender Configuration

- ✓ log4j.appenders.FILE = org.apache.log4j.RollingFileAppender
- ✓ log4j.appenders.FILE.file= e:/loggers/logs.txt
- ✓ log4j.appenders.FILE.MaxFileSize= 2kb
- ✓ log4j.appenders.FILE.MaxBackupIndex= 3
- ✓ log4j.appenders.FILE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appenders.FILE.layout.conversionPattern = %d{yyyy-mm-dd hh:mm:ss } %-c %p : This is %m%n

Log4jTest.java

```
1. package log4app1;
2.
3. import org.apache.log4j.LogManager;
4. import org.apache.log4j.Logger;
5.
6. public class Log4jTest {
7.     static Logger logger = LogManager.getLogger(Log4jTest.class);
8.     public static void main(String[] args) {
9.
10.     logger.trace("trace Message");
11.     logger.debug("debug Message");
12.     logger.info("info Message");
13.     logger.warn("warn Message");
14.     logger.error("Error Message");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
15.    logger.fatal("fatal Message");
16.
17. }
18.}
```

3. DailyRollingFileAppender

The main intention of DailyRollingFileAppender is to roll files in FileAppender on daily basis.

If we want to use DailyRollingFileAppender in log4j applications then we have to use "org.apache.log4j.DailyRollingFileAppender" and we have to use "datePattern" property in properties file.

Note: If we are using DailyRollingFileAppender in our application then it is not required to use "maxFileSize" and "maxBackupIndex" properties in log4j.properties file.

Note: The default pattern for datePattern property is '.' yyyy-MM-dd , it will roll over every day midnight

For "datePattern" property we are able to use the following values.

1. '.' yyyy-MM : It will roll over at the end of each month and at the beginning of the next month.
2. '.' yyyy-MM-dd : It will roll over at midnight each day.
3. '.' yyyy-MM-dd-a :It will roll over at midday and midnight of each day.
4. '.' yyyy-MM-dd-HH : It will roll over at the top of every hour.
5. '.' yyyy-MM-dd-HH-mm : It will roll over every minute.
6. '.' yyyy-ww : It will roll over on the first day of each week depending upon the locale.

Example:

log4j.properties

```
log4j.rootLogger =ALL, FILE
```

#FileAppender Configuration

- ✓ log4j.appender.FILE = org.apache.log4j.DailyRollingFileAppender
- ✓ log4j.appender.FILE.file= e:/loggers/logs.txt
- ✓ log4j.appender.FILE.DatePattern='.' yyyy-MM-dd-HH-mm
- ✓ #log4j.appender.FILE.MaxFileSize= 2kb
- ✓ #log4j.appender.FILE.MaxBackupIndex= 3
- ✓ log4j.appender.FILE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.FILE.layout.conversionPattern = %d{yyyy-mm-dd hh:mm:ss } %-c %p :
This is %m%n

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Log4jTest.java

```
1. package log4japp1;
2.
3. import org.apache.log4j.LogManager;
4. import org.apache.log4j.Logger;
5.
6. public class Log4jTest {
7.     static Logger logger = LogManager.getLogger(Log4jTest.class);
8.     public static void main(String[] args) {
9.
10.
11.     logger.trace("trace Message");
12.     logger.debug("debug Message");
13.     logger.info("info Message");
14.     logger.warn("warn Message");
15.     logger.error("Error Message");
16.     logger.fatal("fatal Message");
17.
18. }
19.}
```

4. ExternallyRolledFileAppender:

This appender listens on a socket on the port specified by the Port property for a "RollOver" message. When such a message is received, the underlying log file is rolled over and an acknowledgment message is sent back to the process initiating the roll over.

Note: ExternallyRolledFileAppender is required Socket Programming and it is deprecated

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Layouts

Layouts:

The main intention of Layout is to define a particular Structer for the logging messages.

There are four types of Layouts existed in Log4j.

1. SimpleLayout
2. PatternLayout
3. HTMLayout
4. XMLLayout

1. Simple Layout:

SimpleLayout is able to prepare logging messages with "LEVEL - Log_Message". Log4j has provided a predefiend class to represent Simple Layout,that is, org.apache.log4j.SimpleLayout

EX:

log4j.properties

- ✓ log4j.rootLogger = ALL, FILE
- ✓ log4j.appender.FILE = org.apache.log4j.FileAppender
- ✓ log4j.appender.FILE.file = E:/loggers/logs.log
- ✓ log4j.appender.FILE.layout = org.apache.log4j.SimpleLayout

Test.java

```
1. package com.durgasoft.log4j;
2. import org.apache.log4j.LogManager;
3. import org.apache.log4j.Logger;
4.
5. public class Log4jTest {
6.     static Logger logger = LogManager.getLogger(Log4jTest.class);
7.     public static void main(String[] args) {
8.         logger.trace("Trace Message");
9.         logger.debug("Debug Message");
10.        logger.info("Info Message");
11.        logger.warn("Warn Message");
12.        logger.error("Error Message");
13.        logger.fatal("Fatal Message");
14.    }
15. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
16.  
17.}
```

OP:

- TRACE - Trace Message
- DEBUG - Debug Message
- INFO - Info Message
- WARN - Warn Message
- ERROR - Error Message
- FATAL - Fatal Message

2. Pattern Layout:

It able to prepare logging messages w.r.t the provided pattern. To represent Pattern Layout, Log4j has provided a predefined class in the form of "org.apache.log4j.PatternLayout".

To define pattern for the logging messages, we must use "ConversionPattern" property from PatternLayout class.

EX:

log4j.properties

- ✓ log4j.rootLogger = ALL, FILE
- ✓ log4j.appender.FILE = org.apache.log4j.FileAppender
- ✓ log4j.appender.FILE.file = E:/loggers/logs.log
- ✓ log4j.appender.FILE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.FILE.layout.conversionPattern = %d{dd/MM/YYYY HH:mm:ss} %-c %p : This Is %m%n

Log4JTest.java

```
1. package com.durgasoft.log4j;  
2.  
3. import org.apache.log4j.LogManager;  
4. import org.apache.log4j.Logger;  
5.  
6. public class Log4JTest {  
7.     static Logger logger = LogManager.getLogger(Log4JTest.class);  
8.     public static void main(String[] args) {  
9.         logger.trace("Trace Message");  
10.        logger.debug("Debug Message");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11.    logger.info("Info Message");
12.    logger.warn("Warn Message");
13.    logger.error("Error Message");
14.    logger.fatal("Fatal Message");
15.
16. }
17.
18.

```

OP:

- ✓ 22/08/2018 18:50:885 com.durgasoft.log4j.Log4jTest TRACE : This Is Trace Message
- ✓ 22/08/2018 18:50:886 com.durgasoft.log4j.Log4jTest DEBUG : This Is Debug Message
- ✓ 22/08/2018 18:50:886 com.durgasoft.log4j.Log4jTest INFO : This Is Info Message
- ✓ 22/08/2018 18:50:886 com.durgasoft.log4j.Log4jTest WARN : This Is Warn Message
- ✓ 22/08/2018 18:50:886 com.durgasoft.log4j.Log4jTest ERROR : This Is Error Message
- ✓ 22/08/2018 18:50:886 com.durgasoft.log4j.Log4jTest FATAL : This Is Fatal Message

Note: We will use these patterns in general in PatternLayout for ConversionPattern property.

- ⊕ %d ----> Date
- ⊕ %c ----> Class to which we are providing Logging
- ⊕ %P ----> Logging Level
- ⊕ %L ----> Line Number
- ⊕ %m ----> Logging Message
- ⊕ %n ----> New Line Character

3. HTML Layout

It able to provide all logging messages in the form of Html file. To represent this Layout Log4j has provided a predefined class in the form of "**org.apache.log4j.HTMLLayout**"

EX:

log4j.properties

- ✓ log4j.rootLogger = ALL, FILE
- ✓ log4j.appender.FILE = org.apache.log4j.FileAppender
- ✓ log4j.appender.FILE.file = E:/loggers/logs.html
- ✓ log4j.appender.FILE.layout = org.apache.log4j.HTMLLayout

Log4jTest.java

```
1. package com.durgasoft.log4j;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2.
3. import org.apache.log4j.LogManager;
4. import org.apache.log4j.Logger;
5.
6. public class Log4jTest {
7.     static Logger logger = LogManager.getLogger(Log4jTest.class);
8.     public static void main(String[] args) {
9.         logger.trace("Trace Message");
10.        logger.debug("Debug Message");
11.        logger.info("Info Message");
12.        logger.warn("Warn Message");
13.        logger.error("Error Message");
14.        logger.fatal("Fatal Message");
15.    }
16.}

```

4. XML Layout

It is able to provide logging messages in the form of XML document. To represent XMLLayout, Log4j has provided a predefined class in the form of "org.apache.log4j.xml.XMLLayout"

EX:

log4j.properties

```

log4j.rootLogger = ALL, FILE
log4j.appender.FILE = org.apache.log4j.FileAppender
log4j.appender.FILE.file = E:/loggers/logs.xml
log4j.appender.FILE.layout = org.apache.log4j.xml.XMLLayout

```

Log4jTest.java

```

1. package com.durgasoft.log4j;
2. import org.apache.log4j.LogManager;
3. import org.apache.log4j.Logger;
4.
5. public class Log4jTest {
6.     static Logger logger = LogManager.getLogger(Log4jTest.class);
7.     public static void main(String[] args) {
8.         logger.trace("Trace Message");
9.         logger.debug("Debug Message");
10.        logger.info("Info Message");
11.        logger.warn("Warn Message");
12.        logger.error("Error Message");

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
13.     logger.fatal("Fatal Message");
14. }
15.}
```

Log4j in JDBC Applications:

log4j.properties

- ✓ log4j.rootLogger = ALL, FILE
- ✓ log4j.appender.FILE = org.apache.log4j.FileAppender
- ✓ log4j.appender.FILE.file = E:/loggers/logs.txt
- ✓ log4j.appender.FILE.append = false
- ✓ log4j.appender.FILE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.FILE.layout.conversionPattern = %d{dd/MM/YYYY HH:mm:ss} %-c %p
%L : %m%n

Test.java

```
1. package com.durgasoft.jdbc;
2.
3. import java.sql.Connection;
4. import java.sql.DriverManager;
5. import java.sql.ResultSet;
6. import java.sql.Statement;
7.
8. import org.apache.log4j.LogManager;
9. import org.apache.log4j.Logger;
10.
11. public class Test {
12. //static Logger logger = LogManager.getLogger(Test.class);
13. static Logger logger = Logger.getLogger(Test.class);
14. public static void main(String[] args) {
15. logger.info("Application Starting Point");
16. Connection con = null;
17. Statement st = null;
18. ResultSet rs = null;
19. try {
20. Class.forName("oracle.jdbc.OracleDriver");
21. logger.info("Driver Loaded");
22. con = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system",
", "durga");
23. if(con == null) {
24. logger.warn("Connection Not Created ");
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
25. }else {
26.     logger.info("Connection Established :" + con);
27. }
28. st = con.createStatement();
29. if(st == null) {
30.     logger.warn("Statement is not created");
31. }else {
32.     logger.info("Statement is Created :" + st);
33. }
34. String query = "select *emp1";
35. rs = st.executeQuery(query);
36. if(rs == null) {
37.     logger.warn("ResultSet is not Created with " + query + "''");
38. }else {
39.     logger.info("ResultSet Object is Created :" + rs);
40. }
41. System.out.println("ENO\tENAME\tESAL\tEADDR");
42. System.out.println("-----");
43. while(rs.next()) {
44.     System.out.print(rs.getInt("ENO") + "\t");
45.     System.out.print(rs.getString("ENAME") + "\t");
46.     System.out.print(rs.getFloat("ESAL") + "\t");
47.     System.out.println(rs.getString("EADDR"));
48. }
49. } catch (Exception e) {
50.     e.printStackTrace();
51.     logger.error("Exception :" + e.toString());
52. }finally {
53.     try {
54.
55.         rs.close();
56.         st.close();
57.         con.close();
58.         logger.info("Closing all Resources");
59.     } catch (Exception e) {
60.         e.printStackTrace();
61.     }
62. }
63. logger.info("End of main()");
64. }
65. }
```

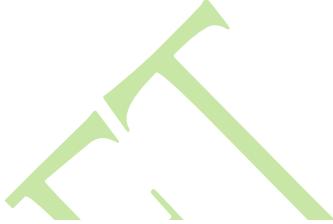
CONTACT US:**Mobile:** +91- 8885 25 26 27 +91- 7207 21 24 27/28**US NUM:** 4433326786**Mail ID:** durgasoftonlinetraining@gmail.com**WEBSITE:** www.durgasoftonline.com**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

Log4j in Web Applications:

1. Keep Log4j1.2.7.jar in web application lib folder.
2. Prepare Configuration File under src folder / under classes folder.
3. It is suggestible to use FileAppenders.
4. In Servlets generate Loggers.

Example:

loginform.html

```

1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Strict//EN" "http://www.w3.org/TR/html4/strict.dtd">
2. <html>
3. <head>
4. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Page</h3>
10. <form method="POST" action=".//login">
11. <table>
12. <tr>
13.   <td>User Name</td>
14.   <td><input type="text" name="uname"/></td>
15. </tr>
16. <tr>
17.   <td>Password</td>
18.   <td><input type="password" name="upwd"/></td>
19. </tr>
20. <tr>
21.   <td><input type="submit" value="Login"/></td>
22. </tr>
23. </table>
24. </form>
25. </body>
26. </html>
```

success.html

```

1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Strict//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

CONTACT US:**Mobile: +91- 8885 25 26 27****+91- 7207 21 24 27/28****US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
2. <html>
3. <head>
4. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status</h3>
10. <h2>
11. Login Success
12. </h2>
13. <h3>
14. <a href=".//loginform.html">|Login Page|</a>
15. </h3>
16. </body>
17. </html>
```

failure.html

```
1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Strict//EN" "http://www.w3.org/TR/html4/strict.dtd">
2. <html>
3. <head>
4. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status</h3>
10. <h2>
11. Login Failure
12. </h2>
13. <h3>
14. <a href=".//loginform.html">|Login Page|</a>
15. </h3>
16. </body>
17. </html>
```

web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.c
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

<!-- XML configuration for Java EE web application -->
<web-app>
    <!-- Application Display Name -->
    <display-name>app4</display-name>
    <!-- Welcome File List -->
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
        <welcome-file>index.htm</welcome-file>
        <welcome-file>index.jsp</welcome-file>
        <welcome-file>default.html</welcome-file>
        <welcome-file>default.htm</welcome-file>
        <welcome-file>default.jsp</welcome-file>
    </welcome-file-list>
    <servlet>
        <!-- Servlet Description -->
        <description></description>
        <!-- Servlet Display Name -->
        <display-name>LoginServlet</display-name>
        <!-- Servlet Name -->
        <servlet-name>LoginServlet</servlet-name>
        <!-- Servlet Class -->
        <servlet-class>com.durgasoft.servlets.LoginServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <!-- Servlet Name -->
        <servlet-name>LoginServlet</servlet-name>
        <!-- URL Pattern -->
        <url-pattern>/login</url-pattern>
    </servlet-mapping>
</web-app>
```

log4j.properties

- ✓ log4j.rootLogger = ALL, FILE
- ✓ log4j.appender.FILE = org.apache.log4j.FileAppender
- ✓ log4j.appender.FILE.file = E:/loggers/logs.txt
- ✓ #log4j.appender.FILE.datePattern = 'dd-MM-yyyy-HH-mm'
- ✓ log4j.appender.FILE.layout = org.apache.log4j.PatternLayout
- ✓ #log4j.appender.FILE.layout = org.apache.log4j.PatternLayout
- ✓ log4j.appender.FILE.layout.conversionPattern = %d{dd/MM/YYYY HH:mm:ss} %-C %p %L : %m%n

LoginServlet.java

```

1. package com.durgasoft.servlets;
2.
3. import java.io.IOException;
4.
5. import javax.servlet.RequestDispatcher;
6. import javax.servlet.ServletException;
7. import javax.servlet.http.HttpServlet;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. import javax.servlet.http.HttpServletRequest;
9. import javax.servlet.http.HttpServletResponse;
10.
11. import org.apache.log4j.Logger;
12.
13. import com.durgasoft.service.UserService;
14. public class LoginServlet extends HttpServlet {
15.     private static final long serialVersionUID = 1L;
16.     public static Logger logger = Logger.getLogger(LoginServlet.class);
17.     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
18.         logger.info("LoginServlet Started-doPost()");
19.         String uname = request.getParameter("uname");
20.         String upwd = request.getParameter("upwd");
21.         logger.info("User Request Received with User Name :" +uname+ " With password :" +upwd);
22.
23.         UserService userService = new UserService();
24.         String status = userService.checkLogin(uname, upwd);
25.
26.         RequestDispatcher reqDispatcher = null;
27.         if(status.equals("success")) {
28.             reqDispatcher = request.getRequestDispatcher("success.html");
29.             reqDispatcher.forward(request, response);
30.         }else {
31.             reqDispatcher = request.getRequestDispatcher("failure.html");
32.             reqDispatcher.forward(request, response);
33.         }
34.         logger.info("Login Status :" +status);
35.         logger.info("End of LoginServlet");
36.     }
37. }
```

UserService.java

```
1. package com.durgasoft.service;
2. import org.apache.log4j.Logger;
3.
4. import com.durgasoft.servlets.LoginServlet;
5.
6. public class UserService {
7.     static Logger logger = Logger.getLogger(UserService.class);
8.     String status = "";
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. public String checkLogin(String uname, String upwd) {  
10.  
11.  
12.    logger.debug("UserService : checkLogin() Method started");  
13.    if(uname.equals("durga") && upwd.equals("durga")) {  
14.        logger.info("UserService :Login Success");  
15.        status = "success";  
16.    }else {  
17.        logger.info("UserService :Login Failure");  
18.        status = "failure";  
19.    }  
20.    logger.info("UserService :End of CheckLogin()");  
21.    return status;  
22.}  
23.}
```

DURGASU

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,