Spring Framework

Saturday, March 10, 2018 12:02 AM

Why Should we use Spring FrameWork ?

Spring has two major features for which we prefer spring more which are..

1. Versatile
2. Non-Invasive

⇒ Versatile

Spring is versatile in nature that is if we are developing a project using some other framework we can develop the same project using spring from the middle so nothing will get impacted.

⇒ Non-Inphasive

Spring is non-invasive in nature that's if we are developing a project currently using spring and we find that spring is not useful for me we can easily remove spring from our project.

Spring Core Module

There are 3 types of classes they are

1. Pojo classes
2. Java Beans
3. Beans or components

Spring core is all about managing dependencies between component or bean classes.
Spring core says to developers that if you want to manage dependencies between classes you have to design your classes based on one design pattern i.e Strategy design pattern.

Strategy design Pattern

- It was came from Gang of four design pattern.

1. Favour *Composition* over *inheritance.*

2. Always design to *interfaces* never design to *concrete classes.*
3. Your code should open for *extension* and close for *modification.*

Hence we concluded that if we create object of another class it is very complex. So we have to give our classes to spring core so it will instantiate the object of our classes.
If we want to give our classes to spring for instantiating object of our classes we have to send the classes in spring understandable format i.e in a xml file which is called ***Spring Bean Configuration file.***
The description that we are providing about our classes to instantiate object and to manage dependencies is called one *bean definition.*

```
<beans>
  <bean id="MessageWritter" class="com.fsc.beans.MessageWritter"/>//bean definition
  <bean id="HtmlMessageConverter" class="com.fsc.beans.HtmlMessageConverter"/>//bean definition
</beans>
```

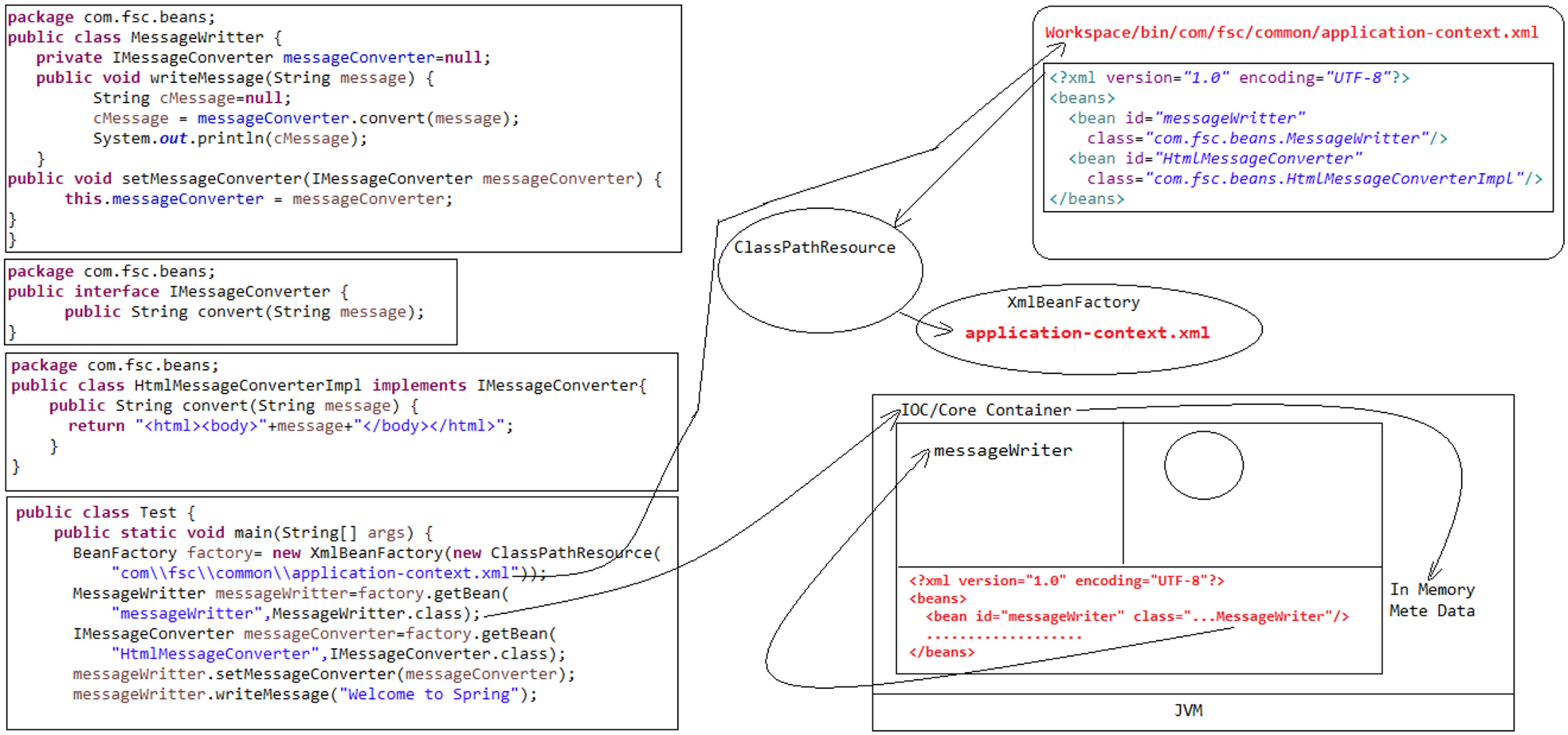
How BeanFactory Creates Object of the bean we defined

```java
package com.fsc.beans;
public class MessageWritter {
   private IMessageConverter messageConverter=null;
   public void writeMessage(String message) {
         String cMessage=null;
         cMessage = messageConverter.convert(message);
         System.out.println(cMessage);
   }
public void setMessageConverter(IMessageConverter messageConverter) {
      this.messageConverter = messageConverter;
}
}
```

```java
package com.fsc.beans;
public interface IMessageConverter {
      public String convert(String message);
}
```

```java
package com.fsc.beans;
public class HtmlMessageConverterImpl implements IMessageConverter{
    public String convert(String message) {
      return "<html><body>"+message+"</body></html>";
    }
}
```

```java
public class Test {
    public static void main(String[] args) {
      BeanFactory factory= new XmlBeanFactory(new ClassPathResource(
          "com\\fsc\\common\\application-context.xml"));
      MessageWritter messageWritter=factory.getBean(
          "messageWritter",MessageWritter.class);
      IMessageConverter messageConverter=factory.getBean(
          "HtmlMessageConverter",IMessageConverter.class);
      messageWritter.setMessageConverter(messageConverter);
      messageWritter.writeMessage("Welcome to Spring");
```



(How BeanFactory creates Object)

beans>

<bean id="first" class="com.ss.beans.FirstClass"></bean>

<bean id="second" class="com.ss.beans.SecondClass"></bean>

</beans>

It is the convention to write the spring-bean-configuration-file. We have to follow the syntax and rule of the dtd and xsd.

By help of id we going to get particular class object.

And class property represent the actual class path location for creating an object.

- If we want to refer an object of the one class to the other class then another tag has been used i.e. property tag.
- It contains name and ref attribute, name is used for
- How To Create IOC Container
- BeanFactory factory=new XmlBeanFactory(new ClassPathResource(RelativePath of SpringBeanConfigurationFile));
- Here 1st ClassPathResource(CPH) object will be created . CPH goes to Resource(Resource is an interface which contain 2 implementation classes CPH & FSR) bz Resource know the physical path location of that xml file.(if you pass the relative path then CPH or if you pass absolutePath then FSR).then Resource read the file and give the inputStream Resource to the xmlbeanfactory.then xmlbeanfactory check wheather the xml is wellformed or not.valid or not. After checking
- The beanFactory create one logical memory partition in the jvm which is called as IOC container or Core container. After creating IOC it divide into 2 parts and put the whole xml file in one part which is called as inMemoryMetaData.(xml contains information about our class so its called as metaData) and give the reference of IOC container to the Factory . Another part of IOC contain the object space.
- When we try to create object of a class means
- MessageWriter messagewriter=factory.getBean("messagewriter",MessageWriter.class);
- When we call factory.getBean() then my factory class does not goes to The physical path location of that file bz it is already available as the part of Ioc container
- It is going to IOC container 1st it check weather the object which it is looking for is already created or not .if it is created then it is just return the reference of that object to factory.if not created
- Then it goes to the logical memory partition read the xml file identifying the bean defination ,get the class name for the corresponding id that we are given.and create the object store that object inside
- the object space as key value,where key is the id and the object as the value and return the reference of the object to the factory.
- class XmlBeanFactory{
- private Resource resource;
- public void getResource(Resource);
- }
- interface Resource{ this code is the internal coding of how Resource works
- input Stream get Resource();
- }
- class ClassPathResource implements Resource{
- public void get Resource(){
- logic
- }
- Class FileSystemResource implements Resource {
- Public void get Resource(){logic}
- }

IOC Principles

Ioc principles helps in collaberating the objects in one place and manage the dependency

It is divided Into two parts

- Dependency Pull

  Dependency Lookup
  Contextual Dependency Lookup

- Dependency Injection

  Setter Injection
  Constructor Injection

Dependency Lookup

If A Class is Completely Dependent on another class object as dependent then it has to go to that class and pull that object from that class ,so my class is tightly coupled on another class

For example in jdbc for persisting the data we need the connection for getting the connection from connection pulling(inside my application server) we need the dataSource object .
dataSource object is the object that is common to all the application so application server put that object inside the JNDI registry.(JNDI registry is the Global Memory place where we need to put our object if that common to all the application)

- Before placing connection object into JNDI registry first we have to create an object and place into the connection pool.
- Using DataSource object we can provide the required info to the application server and server going to create the connection with database. And it will place into the connectionPool.
- So any programmer going to get the connection from the connection pool.

```
Class person{
    public void savePerson(string id ,String Name){
    InitialContext ic = new InitialContext();
    DataSource ds =(DataSource)ic.getConnection("/JNDI Registry Name");
```

.....

.....

}

As my class is dependent on another class object and To get the connection the person class has to lookup into the jndi registry so it is called as Dependency lookup

Contextual Dependency Lookup

If my class want the object of another class as dependent object then my class should go to the class and ask for the object, but that class should know the class of mine then only he can give me the object which I want . So before giving the object that class do one contract with my class and give the object , as it gives the object by write one contract so it is called as contextualdependencylookup.

For exa if my servlet class wants the servletConfig object , for getting the object it has to tell to the servlet container that I want servletConfig object give me that object. (servlet container did not place that object inside the jndi registry because it is private to that corresponding servlet)

so it tell to my servlet that if you want the object then you need to implements your class from my provided servlet interface override one method called init method by passing servletConfig as parameter . So when I create the object of your servlet I can call your init method and I can pass the object as parameter to your init method,

So for my servlet class for getting the object from servlet container it undergoes some contract process that's why it is called as ContextualDepndencyLookup

```
Class Xservlet extends HttpServlet implements Servlet{
    ServletConfig cong;
    public init(ServletConfig config){
    this.config= config;
    }
```

```
    ServletContext context = config.getServletContext();
    ……..//logic to get internal details
}
```

Dependency Injection

If my class wants the other class object as dependent object (either my class should go and create the object or pull the object but when it try to pull or try to create it is become tightly coupled with another class)so it is recommended to go for Dependency Injection

In dependency injection the other class object is injected to my class so it is called dependency injection. It is two types setter & getter injection

Setter Injection

```
<?xml version="1.0" encoding="UTF-8"?>

<bean id="messagewriter" class="com.sc.beans.MessageWriter">
<property name="messageconverter" ref="htmlmesageconverter"/>
</bean>

<bean id="htmlmessageconverter" class="com.sc.beans.HtmlMessageConverterImpl"/>
<bean id="pdfmessageconverter" class="com.sc.beans.PdfMessageConverterImpl"/>

</beans>
```

This is my spring bean configuration file

- **BeanFactory factory=new XmlBeanFactory(new ClassPathResource(RelativePath of SpringBeanConfigurationFile));**

  When we execute this line jvm will create ioc container and goes to the classpath location of that xml file and read all the data , and put all the data inside the ioc container as in memory meta data

  MessageWriter messagewriter=factory.getBean("messagewriter", MessageWriter.class);

  When I execute this line the ioc container goes to the spring bean configuration file and read the bean defination for that id
  1st ioc check wheather the corresponding class for that id is available or not , if available it read the class name and create the object for that class name(if not available it throws an exception)

  After creating the object it check wheather you wrote any property tag or not for that bean defination (property tag defines that, this class contain one attribute and wrote one setter for that attribute so while creating the object , for that attribute read the reference class name and create the object and set this object into that setter)if you write then it hold the object and check for this attribute are you wrote any setter or not , it will not check that wheather the attribute is available or not .

  if available Then ioc container create the object and inject the object to that class by calling the setter so it is called as setter injection

  So hence if my dependent object will injected into the target class object by calling the setter then it is called as setter injection

  Constructor Injection
  If my dependent object will inject into my target class by passing the dependent object as argument to the constructor then this type of injection is called as constructor injection

```
<?xml version="1.0" encoding="UTF-8"?>

<bean id="messagewriter" class="com.sc.beans.MessageWriter">
<constructor-arg  ref="htmlmessageconverter"/>
</bean>

<bean id="htmlmessageconverter" class="com.sc.beans.HtmlMessageConverterImpl"/>
<bean id="pdfmessageconverter" class="com.sc.beans.PdfMessageConverterImpl"/>

</beans>
```

This is the structure of spring bean configuration file in case of spring bean configuration file

IOC CONTAINER(inversion of control)

The full form of ioc container is inversion of control because (as it helps in collaberting the object and it manage the dependency so it is called as ioc container)

If x class wants y class object as an dependent then in general when we manage the dependency
y class become the target class for the class x
X class become the dependent class for the class y

But in spring the dependency management done by ioc container so
If x class wants the object of y class then ioc container inject the object of y class into the x class

So for ioc container x become the target class and y class become the de-pendent class
That's why ioc container is called as inversion of control

Different between constructor ,setter injection

Setter injection

```
Class A{
Public void setA(B b){this.b=b;}
}
Class B{}
```

Here we can create the object of A as
A a=new A();
Because it call default constructor and create the object
After creating the object we can set the dependent object

Different between constructor ,setter injection
1st difference

Constructor injection

```
Class A{
Public class A(B b){this.b=b;}
}

Class B{}
```

Here we cannot create the object as
A a=new A();
Bz A contain B as an argument so in order to create
A we need to pass B as an argument
So incase of constructor injection the dependent object
Will inject to the target class at the time of target class
Object creation

2nd difference
Constructor injection
Here when we try to create the object ioc container read the
Bean defination and and it check the constructor-arg and
Take the reference class 1st create the object of that class
Then create the object of the target class by passing this
Dependent object as an argument to the constructor of that
Target class

```
Class A{
Public class A(B b){this.b=b;}
}

Class B{}
```

Reason when should we use constructor or injection

- **If you want the dependent object will be available at the time of target class object creation then we should go for constructor injection**
  As it is creating the object of dependent at the same time when ioc container create the object of target class
- **If you don't want the dependent object available at the target class object creation then go for setter**

Setter injection

```
Class A{
Public void setA(B b){this.b=b;}
}

Class B{}
```

A a=new A();
Because it call default constructor and create the object
After creating the object we can set the dependent object

So incase of setter injection the dependent object is inject into the target class after creating the object of the target class

But here ioc container directly create the object of the target class by calling
The default constructor (which is added by compiler) and then set the dependent object

Target class

Class B{}

Reason when should we use constructor or injection

Here when we try to create the object of class A we must have The object of B

1

So we can call

A a=new A(new B);

So in case of constructor injection it is mandatory that the dependent object should be injected

Here ioc container don't call the default constructor bz we place the parameterized constructor so for creating the object we must have the dependent object

Reason when we go for what

If your target class don't work without the dependent object then you should go for constructor injection

In our example MessageWriter don't work without the dependent object Htmlmessageconverterimpl so it is recommended to go for constructor injection

Class B{}

Here for creating the object of A we don't need B

We can create A a=new A();

We can set as a.setA(B b);

In case of setter injection it is not mandatory that the dependent object should injected for creating the target class object creation

Reason for setter

If without the dependent object my target class will work

And we can create the object of target class then it is recommended to go for setter injection

And after creating the object you can set the dependent object any time

3rd difference

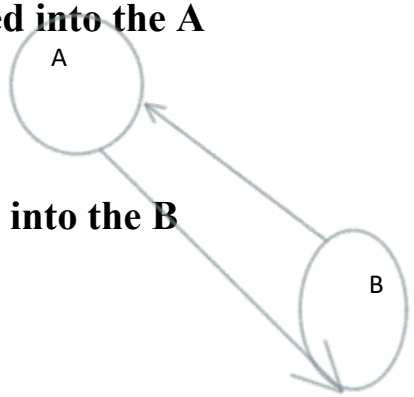
If your classes are cyclic dependent on each other then it is recomeneded to go for setter injection bz by using constructor injection you cant perform the operation

It means in constructor injection when we create the object ioc container 1st check the bean defination for that id and pick the corresponding class name and stop

Then it take the 2 class name (class name for that id and constructor-arg class) as node and plot one graph inside the ioc container

Class A{
Public class A(B b){this.b=b;} **here B is the dependent and it is injected into the A**
}

Class B{
Public class B(A a){this.a=a;} **here A is the dependent and it is injected into the B**
}




```
<bean id ="a" class="A">
<constructor-arg ref="a">
</bean>
<bean id="b" class="B">
<constructor-arg ref="a">
</bean>
```

After plotting the graph it check wheather it is cyclic dependent or not if cyclic it is not going to create the object and stop by throwing exception

Primitive Dependency Injection

Before we are injected object as dependent into the target class , we can inject premitive value as an dependent into the target class
Through setter or constructor injection

Tag for setter `<property name="a" value="Kartik">`
Tag for constructor `<constructor-arg value="Kartik">`

Collection Dependency Injection

We can inject collection of values as an dependent to target class through setter or constructor injection which is called as Collection Dependency Injection

- **Spring supports 4 type of collection. Means it provides 4 tags for these collection**
- **Which means if we don't have the tags we can create the object of collection but we can create empty collection object only. We can set as**
- **<property name="class" ref="java.util.ArrayList">**
- **Because every class which can be create using new operator that class object can create by ioc container so it can create the collection class object also but with empty object**
- **But it create empty collection object but empty collection has no meaning so Spring provide 4 tag for 4 collection Set,List,Map,Properties**
- **syntax for tags are**

```
<property name="subject">
 <list>
 <value>Java</value>                    Setter Injection for List
 <value>DataStructure</value>
 </list>
```

```
</property>

<property name="faculty">
  <set>
    <value>Kartik</value>                    Setter Injection for Set
    <value>Bibek</value>
  </set>
</property>
</bean>

<bean id="college" class="com.cdi.beans.College">
<constructor-arg>
 <map >
   <entry key="Prem">
   <ref bean="course"/>                                  Constructor Injection for Map
   </entry>                                              In 2 way we can write
   <entry key= "Prince" value-ref="course"/>
 </map>
</constructor-arg>

<property name="toppers">
  <props>
    <prop key="Papun">
    MCA
    </prop>
    <prop key="Mamun">
    B-tech
    </prop>
  </props>
</property>
</bean>
```

Before we are using the non generic collection but it is recommended to go for generic collection like

```
<property name="subject">
  <list  value-type="java.lang.String">
  <value>Java</value>
  <value>DataStructure</value>
  </list>
</property>
```

Setter Injection for List using generic collection

Similarly for map also we can use generic

```
<bean id="college" class="com.cdi.beans.College">
<constructor-arg>
  <map  key-type="java.lang.String" value-type="com.cdi.beans.Course">
    <entry key="Prem">
    <ref bean="course"/>
    </entry>
    <entry key= "Prince" value-ref="course"/>
  </map>
</constructor-arg>
```

Constructor Injection for Map
In 2 way we can write
if key is premitive then key-type="java.lang.String"
if value is object type then value-type="fully qualified name of that object"

- **Whenever we create one collection object as dependent by direct using the tags then we don't know that internally ioc container create which type of collection bz we did not specify any collection to create(means for List 4 collection are there like LinkedList,ArrayList, Vector) it is off to ioc container it may create al or may ll.**

- **If two bean class wants to used one attribute as an dependent in there class then we need to write that attribute as dependent for both bean , then our code should be duplicated(means if we using normal tag the collection of value object we cant assign for both the bean by writing separate and using inside the beans)**
- **To overcome these two problems Spring provide one concept called UtilNameSpace which is like as below**

```
<util:list id="col" value-type="java.lang.String" list-class="java.util.ArrayList">
  <value>There are Two team must be there</value>
  <value>20 over match should conduct</value>
</util:list>

<bean id="game" class="com.cdi.beans.Game">
<property name="rules" ref="col"/>
</bean>

<bean id="iccBoard" class="com.cdi.beans.IccBoard">
<constructor-arg ref="col"/>
</bean>
</beans>
```

Constructor confusion

We can solve the constructor confusion by 3 way

- **Type**
- **Index**
- **Name**

Bean Alias

Providing multiple name to our bean defination is called bean-alias

```
<bean id="Kartik" name="prem"/>
<alias name="Kartik" alias="papun"/>
```

Both the way we can give the alias name but in 1st way , acts as an separator

Exa kartik,sahoo this is considered as 2 name if you want to include , also in your name then go for alias tag

BEAN AUTO-WIRING

If we don't want to manage the dependency by me , I want IOC container to automatically identify the dependency and manage the dependency by itself then it is called as bean-auto wiring .

By default auto-wiring is turned off , so to achieve auto-wiring we have to turn on auto-wiring by writing one attribute called autioWire=" mode" mode is of 3 types

1. byName
2. byType
3. constructor

1. byName

byName tells that identify the dependent and manage the dependency by calling setter of the target class on the basis of attribute name

when are telling to ioc automatically manage the dependency then we have to tell to the ioc in which way

we want to manage the dependency by telling the mode of auto-wiring .

For example

```
<bean id="product" class="Product" autioWire="byName"/>

Product product=factory.getBean("product", Product.class);
```

Here beanFactory 1st goes to bean defination pick the bean id then goes to the ioc container search wheather the object is available or not for this bean ,if not then it goes to the inMemoryMetaData pick the class name search the class available or not ,if available then it check wheather you wrote any constructor or not for performing circular dependency check , if not found then it check wheather you enable the autowire or not if you enable.

Then it reads the mode of auto wiring byName then it create the object of product bean and goes to the product class search for which attribute you want to manage the dependency , then it check wheather you write any setter or not for this attribute ,(if not it don't manage the dependency) if you wrote then it picks the attribute name and goes to the ioc container match the attribute name with the bean id that are present in ioc.

If match found then that bean id object it inject to the target class and manage the dependency .

byType

byType tells identify the dependent and manage it by calling the setter of target class on the basis of attribute type

For example

```
<bean id="product" class="Product" autioWire="byType"/>

Product product=factory.getBean("product", Product.class);
```

Here beanFactory 1st goes to bean defination pick the bean id then goes to the ioc container search wheather the object is available or not for this bean ,if not then it goes to the inMemoryMetaData pick the class name search the class available or not ,if available then it check wheather you wrote any constructor or not for performing circular dependency check , if not found then it check wheather you enable the autowire or not . If you enable.

Then it reads the mode of auto wiring byType then it create the object of product bean and goes to the product class search for which attribute you want to manage the dependency , then it check wheather you write any setter or not for this attribute ,(if not it don't manage the dependency) if you wrote then it reads the attribute type and goes to the ioc container match the attribute type with the bean class type that are present in ioc.

If match found then that bean class type object will inject to the target class and manage the dependency .

Constructor

It tells to the ioc container the manage the dependency by calling the constructor on the basis of argument type

For example

```
<bean id="product"  class="Product"  autioWire="constructor"/>
```

```
Product product=factory.getBean("product", Product.class);
```

Here beanFactory 1st goes to bean defination pick the bean id then goes to the ioc container search wheather the object is available or not for this bean ,if not then it goes to the inMemoryMetaData pick the class name search the class available or not ,if available then it check wheather you wrote any constructor or not for performing circular dependency check , if not found then it check wheather you enable the autowire or not . If you enable.

Then it reads mode of auto wiring constructor then it doesn't create the object goes to product class search wheather you wrote any constructor or not (if not it throws exception) if found then it goes to the max parameter ,(for example 3 parameter .2 parameter constructor is there ,it 1st goes to 3 if not suitable then 2) and picks the argument type one by one (if multiple parameter is there) goes to the ioc container match the argument type with the bean class type if found then it inject into the target class

Limitation in auto wire

- ✓ **When we go for byName auto wire then our dependent can easily be broken because in project multiple developers are working , there may be chance that someone may change the bean id accidentally ,if they think that may no one should not use this , so in case of byName auto wire data consistency is not there.**

- ✓ **When we go for byType auto wire it is mandatory that there should be one bean class type for one configuration , because if the bean class type is repeated then ioc will confuse which bean class type to refer , it occurs when if one developer want to give new data to the same class type then it might chance that he may be modify the code.**

- ✓ **When we go for auto wire it is so difficult to understand that which dependent is to be injected just by looking the configuration file, for understand we need to compare the target class with the configuration file . And it take more time so for these reason it is not recommended to use bean auto wire .**

Question ---why spring give auto wire when it is not recommend to use

It is use in poc type project (poc=it means in company whenever we want to understand how to write the logic in the project we are not write the code direct in the project bz if there is some mistake the project may be shut down , then we are going for poc project , in this for specific area we are going to write the code by taking few classes , then for temporary project why we spend our time in writing the configuration file , rather than we are going to use bean auto wire so that our time is taking less) . Poc is a temporary project so we are using bean auto wire , in this I only write there is no chance of misinterpreting .

Working with null

- ✓ While working with constructor injection , it is mandatory to inject the dependent , so in this case if we have nothing to inject as dependent in that case we can give null value to the dependent .

```
Syntax-------<constructor-arg >
                <null/>
              </constructor-arg>
```

Nested Bean Factory

- ❖ While working with multiple ioc container inside the project , if we want to insert one ioc container inside the other ioc container , then we go for nested bean factory. So that child ioc holds the reference of the parent
- ❖ In nested bean factory the one who insert into another one is called as parent ioc container , as it is parent so it must be there before the child one.
- ❖ The other one is called child ioc container
- ❖ In case of nested bean factory the child ioc container should see the beans of the parent ioc container but the parent can not see the beans of the child ioc container .

Syntax for how to insert one ioc into the other

```
BeanFactory   parentFactory=new XmlBeanFactory(new ClassPathResource("parent-beans.xml"));
BeanFactory   childFactory=new XmlBeanFactory(new ClassPathResource("parent-beans.xml"),parentFactory);
```

For the configuration

<ref parent=""/> **it search for the beans in the parent ioc container**
<ref local=""/> **it search for the beans in the local ioc**
<ref bean=""/> **it search first inside the local ioc if available then inject , if not then it go to the parent ioc**

Bean Inheritance

If you want to reuse the configuration of one bean defination inside another bean defination then go for bean inheritance

For achieving the bean inheritance you need to write one attribute in the bean tag level

```
<bean id="car" class="Car" >
      <property name="carNo" value="1"/>
      <property name="model" value="swift"/>
      <property name="color" value="red"/>
       </bean>

  <bean id="car1" class="com.bi.beans.Car" parent="car">
      <property name="carNo" value="2"/>
      <property name="color" value="white"/>
  </bean>
```

When we go for bean inheritance then our configuration must be compatible with the parent configuration

When we go for bean inheritance then we need to make our parent bean defination as abstract bean , bz if we don't make abstract , if any change will happen to the parent bean then it will affect the all the bean defination

which are inherit from that bean

We need to write the attribute abstract="true" , when we make our bean as abstract then we don't need to specify the class name for that bean bz by defining abstract we are telling to ioc to not create the object for that bean

- ✓ So we can specify the bean defination of an interface by making the bean defination as abstract, if ioc not going to create the object then its not problem to define interface in configuration file

```
<bean id="car"  abstract="true">
      <property name="carNo" value="1"/>
      <property name="model" value="swift"/>
      <property name="color" value="red"/>
  </bean>

  <bean id="car1" class="com.bi.beans.Car" parent="car">
      <property name="carNo" value="2"/>
      <property name="color" value="white"/>
  </bean>
```

While working with constructor injection we need to use name attribute

```
<bean id="car"  abstract="true">
      <constructor-arg name="carNo" value="1"/>
      <constructor-arg name="model" value="baleno"/>
       <constructor-arg name="color" value="red"/>
       </bean>

  <bean id="car2" class="com.bi.beans.Car" parent="car">
      <constructor-arg name="carNo" value="2"/>
      <constructor-arg name="color" value="white"/>
  </bean>
```

Inner Bean

Writing one bean defination inside the another bean defination is called inner bean

For exa bicycle is a class which want to use chain class then in general we define both the class as 2 bean ,
But chain is particular to that bicycle only so we don't need to define separately , we can define those like

```
<bean id="bicycle" class="com.ib.beans.BiCycle">
    <property name="company" value="Atlas"/>
    <property name="chain">
      <bean class="com.ib.beans.Chain">
        <property name="chainNo" value="1"/>
        <property name="chainType" value="metal"/>
        <property name="length" value="100"/>
      </bean>
    </property>
 </bean>
```

If one bean require the configuration of another bean , and if the bean particular to that bean ,no one use that bean then define that bean as inner bean.
As it is particular to that bean then we don't need to define the bean id for the inner bean

Collection merging

Collection merging comes into picture when we are performing bean inheritance
In this we merge the one collection of values of one bean inside the other bean by writing one attribute merge="true" in the collection tag level

```
<bean id="btech1stsemmech" class="com.cm.beans.Course">
    <property name="subjects">
```

```
        <list value-type="java.lang.String">
          <value>Mechanics</value>
          <value>ThermoDynamics</value>
          <value>BasicElectronics</value>
        </list>
      </property>
    </bean>

    <bean id="btech1stsemece" class="com.cm.beans.Course" parent="btech1stsemmech">
      <property name="subjects">
        <list value-type="java.lang.String" merge="true">
          <value>ElectronicCommunication</value>
        </list>
      </property>
    </bean>
```

When we go for collection merging we need to ensure that the collection type and the generic type should compatible

IDREF

- **For runtime dependency injection we should go for dependency pull. Bz dependency injection doesn't support runtime dependency. in Dependency injection we can perform injection at design time only**
- **When we go for dependency pull in the configuration if we pass the bean id of another bean defination as value as dependent then it is work but it might chance to face two problems.**
- **1st problem is we cant say by just looking at the configuration that the value is a bean id of another bean defination or not. Bz it is looking like the normal value. So might chance that some one think that It is a normal value , so he can modify the bean id as it is not used by any other . When he modify the bean id as per his requirement then the configuration goes into inconsistent state. And at the time of creating object ioc also not going to check bz we didn't tell to ioc that this value is a bean**
- **When we are going to create the object of that bean ioc create the object with wrong configuration . And at**

the time of middle of the program due to the wrong input the program will go to runtime exception. Abnormally our program will terminate.

- To over come this problem use idref as

```
<bean id="car" class="com.idref.beans.Car">
    <constructor-arg >
    <idref bean="bmwCar"/>
    </constructor-arg>
</bean>
<bean id="audiCar" class="com.idref.beans.AudiEngineImpl"/>
<bean id="bmwCar" class="com.idref.beans.BMWEngineImpl"/>
```

- Here ioc container will check at the time of object creation wheather the bean is available or not . If available then create , if not then it throws exception before starting execution

Bean Scope

Single Ton Class

- If I allow to create one object for a class in entire my application then this class is called as singleTon class . If 3 classes in my application want the object of a class then if all the 3 are allow to create one one object of that class then this class is not single ton,

- So to stop the classes for creating the object of Processor classes we need to make the constructor as private .(so no one can create the object of Processor class).

Que- how to stop a class to participate in inheritance without using final key-word

Make the constructor of that class as private so no one cant create the object. The class which as private constructor that class by default is abstract final class.

- As no one cant create the object then Processor class method should create the object so that write one

method with Processor as return type not void bz this object is used by other class. But to get the object we need to call the method , but to call the method we need object. but we don't have the object as it contain private constructor we cant call the constructor to create the object. So we have to call the method without the object then make the method as static and call it by the class name.

```
Class Processor{
    private Processor(){

    }
Public static Processor createProcessor(){
   return Processor
    }
}

Processor p1=Processor. CreateProcessor();
Processor p2=Processor. CreateProcessor();
```

but with this we can create as many object we want so it is not singleTon class.

- So we shouldn't return the object of the processor. whenever we create store the object in a variable and whenever required , 1st check wheather it is created or not ,if not then create and store in the variable , return that variable , if already create then just return the variable. As we cant store static method object inside the non-static variable so the variable must be static.

```
class Processor{

private static Processor instance;

 private Processor{
      // no op
   }
```

```
public static Processor getInstance(){

      if(instance==null){
            instance=new Processor();
          }
            return instance;
    }
}
```

Question ---why we go for thread

Ans----when the program is running in normal class then if 50 lines of code one part , 50 lines of code another part. to be execute then the two part executed linearly. In 1st part let us take in 6 th line some io operation code are there , so it will not execute by the cpu , it is execute by dmu(dynamic memory unit) at that time the cpu becomes suspent that is ideal time for cpu. So after dmu cpu will execute. So if we go for thread if cpu pause in one thread then it goes to another thread and execute.

When to go for thread----------If in our application we have multiple independent part of code which are going to be execute independently then we have to place that code in different type of thread and should execute concurrently.

When to go for Runnable(i)----------if we have one big job which contain lot of code and which can execute independently then don't go for thread , bz if the object is different then go for thread , but if the object is same why to create the same object for all the thread . Go for runnable, create the object for the runnable. And give the same object to all the thread.

Why we cant call start() 2 times for a thread-------bz start() not only call the run() but it will maintain the status of the thread .let us assume When we create the thread object and call the start() then it create the thread with that object reference , and when we call 2nd time start() it will create another thread with that object reference , then if we call isAlive() then it confused which thread status you are looking for. So when we call 2 times start() for a thread it will throws an exception illegal State Exception.

Execution flow of Runnable(i)------

```
 class Work implements Runnable{
        public void run(){
           //logic
     }
}
Class Test {
    psv main(String[] args){
        Work wr=new Work();
        Thread t1=new Thread(wr);------Line1
        Thread t2=new Thread(wr);
    }
}
```

When we call line1 then thread scheduler goes to thread class run() method not runnable(i) bz it don't know about runnable. Inside the run() of thread class check the condition ad then call the run() of the corresponding runnable class run() method.

The thread class is looks like as

```
Class Thread{
Runnable runnable;
     run();
     Runnable(Runnable runnable){}
     void run(){
         if(Runnable!=null){
            Runnable.run();
         }
    }
}
```

ClassLoader

- When we are execute a java class it converted into .class file which contain byte code , jvm execute the byte code , but jvm don't go to the location and read the file , classLoader goes to the physical file system and read the .class file load into the jvm .
- class loader are also the java classes . It is of 3 types BootStrap classLoader , Extension classLoader , System classLoader /application classLoader
- All the 3 classLoader doesn't read class file from any where , they only reads from predefined location .at start time they don't load all the class they load the class on demand.
- BootStrap classLoader is native , it is platform dependent , it is not java class , that's why java has multiple version of os , for different os multiple version java is there , it reads only from JRE_HOME\lib\rt.jar
- Extension classLoader is the immediate sub-class of Boot-strap class loader , it reads from JRE_HOME\lib \ext and from system property java.ext.dirs directory. Here all the external files are there .
- If we want scope of a variable inside the method then declare in side the method, if in the class then declare as attribute , but if we want the scope of variable inside my application for all the classes then declare as public static.
- If we want the scope across the application means for the jvm then we need to go for system property . Syntax for declare a jvm level variable is java -Di=10 -Dj=20 className . When we are define the scope across the jvm then it value only treated as string only. String m=System.getProperty(i) here 10 come if we replace j then20 will come
- Whenever jvm request to classLoader for loading the class , it doesn't load the bytecode , 1st it create the CLASS object for my class. The CLASS object contain the structure of my class(means the method ,variable etc). The CLASS looks like as

```
class CLASS{
   Private static final class;
   Private method[] methods;
   Private field[] fields;
   Private classLoader classLoader;
```

}

- We can know which classLoader loads our class (suppose for A class) sop(" A.class.getClassLoader().getName()")
- In two way we can know wheather extension classLoader loads our class or not 1st way is create the jar file for the class you want to check , then put that jar file in your ext directory where your java is install , then run your class by printing sop(" A.class.getClassLoader().getClass().getName()")
- The 2nd way is java -Djava.ext.dirs=fileLocation ClassName
- There are 3 ways to check for application class loader 1st way is run set CLASSPATH=file location then run the class
- The 2 nd way is java -cp fileLocation className or java -classpath filelocation className
- The jar is of 2 types distribution jar and executable jar .
- The distribution jar are the normal jar . If we are giving this jar to our client the client has to know how to run our classes . But in real the client don't know how to run the application. So it is recommended that we have to make this jar as executable jar by putting manifest file inside the jar. Which can run without knowing the classes information.
- 1st compile the dependent class , create the jar , then put inside the target class folder , then set class path to the jar and then compile the target class, then create one text file named it manifest then inside the manifest write two attribute Main-Class: qualified name and Class-Path: jar name pressenter then create the jar jar -cvfm jarname manifest.txt classname.

Principles of class loader

- Principles of Delegations
- Principles of Visibility
- Principles of Uniqueness

Principles of Delegations

Whenever jvm request to load the class then 1st it goes to the list class loader from the hierchy i.e system class loader , but system class loader doesn't try to load the class it delegate to the parent means extension class loader ,

extension delegate to the bootstrap class loader , then bootstrap try to load , if available load otherwise it return the request to extension , if here also not available request comes to system class loader . This is called as delegation.

How to prove delegation 1st create ,compile the class then create the jar for that class , put the jar file in the ext directory, set the class path to the .class file then run that class .as the jar file is in the ext directory so the class loader will be the extension class loader

Principles of Visibility

If a class is load by one class loader then it can see the class that are load by the same class loader or the class that are load by the parent class loader.it can not see the class that are load by the different class loader.

How to prove visibility write 2 class one is dependent and the other is target, 1st compile the dependent then the target , after put the dependent .class folder outside and create the jar for the target class . Then put that jar in the ext directory. Then delete the .class file of target class , then put the dependent class file inside the package directory. Then run the target class . Exception will arise bz as it is load by extension loader but the dependent class is load by the system class loader bz it is under the class path. Just reverse the mechanism it work.

Principles of Uniqueness

No classes is load by two class loader (means if one class is load by one class loader then the other class loader cannot load the class again)

How to prove uniqueness cerate one class in which write one method , create another main method class then create the object and call the method of that class , after then get the class loader of the main class , then this reference to the other class by calling class.forName.

- **We can get the reference of system, extension class loader but we cant get the bootstrap class loader reference. If we try then we get null. By calling A.class.getCgetClassLoader().getParent().getParent();**

- As we can not get the reference of the bootstrap class loader we cannot replace the core jdk classes with our classes that are inside the bootstrap class loader.

Why java.lang is called reserved package java.lang is not reserved package , whenever we are create our class under the java.lang package (exa String class we take)then our class gets compiled but if we are trying to create the object of our class then jvm tells to the system class loader , according to the principle of delegation the request forward to the boot strap as , inside the bootstrap already the String class is available so it loads the core jdk classes , so there is no chance that our class get loaded and executed.

Class loader name space

If we try to execute one class then it load by the extension class loader if it is in ext directory, assume that we are creating our own custom class loader , so we can break the principles of class loader, as we break the rule so it doesn't delegate the request so it again load by our custom class loader , then inside the jvm not one copy of that class will be load, there are 2copy of that class will be load as it load by 2 class loader,

But the problem is whenever we are trying to execute the class , then there are 2 class copy of same class name, to avoid this problem every class loader at the time of loading the byte code into the jvm it holds the reference of the class loader with the class name. this is called as class loader name space.

Difference between ClassNotFoundException & noClassDefFoundError

In java in 2 way we can load the class

A a=new A(); this is called implicit class loading
Class,forName(args[0]); this is called as explicit loading

If we try to execute one class by implicit class loading then at the compilation time the class name is bounded to that class after the compilation if at runtime the class defination is not found from the .class file then noClassDefinationFoundError will arise . When this error will arise Then we don't need to evaluate our code to solve the error bz if our code is wrong then it doesn't get compiled . We need to check our class path.

When ClassNotFoundEception arise then we need to check our code as well as the class path bz

Phases of ClassLoader

- Loading
- Linking
  - -----Verifying
  - -----Preparing
  - -----Resolving
- Initializing

Loading

In the loading phase class loader convert the package name to corresponding directory means the . Is divided the name to sub directory . And class loader goes to the corresponding file location and load the byte code into an byte array. You can say that it dumped the byte code in the jvm.

Verifying

In this phase it checks wheather the .class file is modified or not. Means at the compilation time for the byte code of the class ,java compiler generate one unique no. when class loader try to load the class at the time of verifying stage class loader also generate one unique no , if that no is matched with the compiler generated code then the .class file is not modified if changed then it throws the exception java.lang.ClassFormatError: Illegal UTF8 string in constant pool in class file com/phase/Launcher

In this stage it also check forward compatibility check(if your class is compiled in 1.5 version then it can run in up to 1.9 version but the class is compiled in 1.4 cant run in 1.9 this is called forward compatibility check)

Also forward port compatibility check is done(if one class is compiled in 1.8 version then it cant run in 1.7 version) . If both the check is failed then it throw exception .classBadVersionError.

Preparing

In this stage the skeleton representation of the byte code is created means the Class object created for the class

Resolving

In this stage it check wheather the class is extend from other class is or not if extend then it load the parent class , if parent is unavailable then it show **Could not find or load main class Launcher** . If we refer other class as attribute inside the class then it load the current class without throwing exception even though the attribute class is deleted .

You can check after compile the classes delete the parent class then run the subclass

Initializing

In this phase the class loader the static variable is allocate memory and initialized by the class loader, static block get executed by the class loader. If you want to stop executing static block you can stop

```
ClassLoader cls=B.class.getClassLoader();
Class ab=Class.forName("A",false,cls);
```

J2EE Environment ClassLoader

- ✓ As there are classLoader in the jdk environment , similarly in the j2ee environment also the classLoader concept is there , in the j2ee environment the classes are not load by the jdk classLoader (bootstrap,extension,system) .

- ✓ The jdk classLoader are not load the class but this classloader has to be present . Bz the core jdk classes that are present inside the rt.jar are used in j2ee enviornment also .
- ✓ So in the j2ee enviornment the class are load by some other class loader. That class loader are provided by the application server vendor who are provide the server(tomcat, weblogic ,glashfish).the class loader name is differ from server to server.
- ✓ But the way how to load the classes and where to load the classes that are provided by the sun micro system. There is one specific location where the classloader are going to load the classes. There are some fixed structure that every vendor has to follow(web-apps is the fixed location). This class loader are the immediate child of system class loader.
- ✓ Whenever the request goes to our servlet , then internally it goes to servlet container , servlet container calls the service method . Servlet container doesn't execute , it internally calls jvm to execute. Jvm goes to th class loader ask for loading the class.

War

- ✓ In our servlet every application are one war packaging model. For every war file the server create one class loader at the time of deployment . That class loader are dedicated to that war file only. And they are maintaining in one directory. For every war file the corresponding class loader is there.
- ✓ But when the request goes to jvm , it goes to the system class loader as it is the list class loader in the hierchy . But in j2ee environment there are one class loader in the list hierchy for every server. But jvm don't go to that class loader.
- ✓ So here the servlet container 1st pick the application name and find the corresponding class loader and tells to the jvm to ask this class loader has to load the class by passing in the method Class cls=Class.forName("A",true,Class loader name); .
- ✓ So as every application is load by one class loader if 2 war file deploy in one jvm then the classes that are load by one class loader that cant see the classes that are load by another class loader. Due to the principle of visibility. As the classes are loaded by the siblings class loader.
- ✓ So the singleton classes that are loaded by 2 war can create 2 object as it is load by 2 class loader, there are 2 references are there in the same jvm. The scope of singleton classes is over the hierchy of the class loader.
- ✓ If one variable is declare as public static it also cant access by other classes outside the application.

Ear

- ✓ The problem is if one class is common to 2 application we have to load 2 times that class as the 2 application loaded by 2 class loader no one can see each other.so the jvm memory is wasted.
- ✓ So here ear comes into picture we can place 2 war file in the ear and the classes that are common to the both application we can put in the ear file. In ear the classes are loaded by separate class loader , in general it called application class loader
- ✓ The information about the war file are write inside the application.xml.

Ear

APP-INF
  classes
  lib
META-INF
  application.xml
app1.war
app2.war

- ✓ The classes that are common to both the application put that classes in the classes folder or lib folder so that it can be loaded by application class loader and the classes that are loaded by other class loader can see this classes.
- ✓ Again in the ear the war file are loaded by other class loader and that are child class loader of the application class loader. Means app1 .war and app2.war are loaded by different class loader and this classes can see the classes that are loaded by application class loader.
- ✓ If we want to share some classes that are common to the ear then we can not do. So some of the application server vender are giving the solution for this that is Domain comes into picture.
- ✓ We can place the ear inside the domain and the common classes in the domain level. Then this classes is throughout all the ear. As it is loaded by one class loader called domain classLoader , which is the parent of the application loader and war class loader.

When to use singleton class

There are 3 situation where we can make a class as singleton class

- If a class doesn't have any attribute ,so that the class doesn't have any state , so for using the class we can call the method of that class , if I call 4 times I need to refer the object of that class , but every time the object same bz it doesn't contain any state as the object doesn't have any state so we don't need to create multiple object for that class ,

```
Class A{
//no attribute
Public void m1(parameter){}
}
```

we can call this class by passing value to the parameter , so we don't need to depend on the state of the object to call the class as we are passing parameter . In this class if I create multiple object for that class then it has no meaning as all the time it doesn't contain any state.

- If a class contain attribute and the attribute contain read only value(if the value is fixed which is every time same)then the state of the object is same , as many number of times we are creating the object then in this case we shouldn't create the multiple object for that class.

```
Class A{
Private string pi=3.142;
    Public void m1(){}
}
```

Here the pi value is same for every time when we are creating multiple object so in this case we should create one object for that class.

- Question ------how the servlet request and the response is managed in j2ee enviornment .

- **When the end user hit the jsp page then actually in real the request comes to the servlet , but it is not true bz the request comes to the servlet container. actually the request comes over the network , so if the servlet receive the request then the servlet class should contain the network logic . If 10 servlet classes is there then we have to write the network logic in all the classes.so to avoid this boiler plate logic servlet container receives the request and internally it deals with the network logic , so we don't need to worry for network logic**
- **After receive the request the servlet container read the data and put the data in a http servlet request object and pass the data to the servlet(bz id doesn't know how to process the data) . But pass to the servlet it has to call the servlet class method . So it require the object of the servlet class. So it tells to us that create your class by extending from http servlet so that I can get the object of your class and can call the service method of your class.**
- **In our application there are multiple servlet class is there. to know the information about our class we are configuring the web.xml in our application.**
- **When we send a request the servlet container receive the request and goes to the url takes the context root and goes to the corresponding war of the xml file, and search the class name with the servlet path and call the servlet class.**

Question ----------Without Making the class as singleton class we can make all the method as static if the class doesn't have any state. Which one is mora preferable .

Answer ----------we can make all the method as static , as the class doesn't have any state so the object of the class is not required. By simply making all the method as static we don't need the object of the class to call the static method of the class, but we cant stop to creating the object of the class, if one mad person is there he can create unnecessarily multiple object.

- **Then the interviewer will tell that we can make this class as abstract class, so that we can stop creating multiple object of the class. then we have to tell**

- **We can stop creating multiple object by making the class as abstract class but we cannot enhance the logic of the class if we want, bz all the method are static which cannot participate in inheritance bz static method cannot be over ridden.**

- Let us take one real world application in which we are going to know about how to build an application
- The application is write registration application for end user.in foodpanda order.

- when we are write the 1st page of jsp page in which we are writing the registration form , for the registration we are providing box for submitting their data, so we are also providing the select control form to the end user.
- Right now if I am providing city ,state, country for some limited area. But those value are not fix bz if business is successful then we may have to add some more city or state to our application.
- Then if we are hard code this value in the jsp page direct then when we want to add then it face the problem as it is not written in one single page.
- 1st problem is to make the changes is difficult as it is not written in one single page
- 2nd is to find where we are going to modify we need to check which is difficult
- 3rd is if we are modify in some place and we are not modify in some place then my application goes to inconsistent state.

Types of data base table

- Master table
- Operational table
- Transactional table

Master

In our application if some data has to be inserted before the application running (means without this data our application are not going to work) this type of data table are called master table

Operational

If the data is needed at the runtime this type of data table storing is called operational table

Why we use JSP

- ✓ **In a html page we can show only static input of data. We cant show dynamic input data. So if we want to show dynamic input data then we have to write the view data logic inside the servlet.**
- ✓ **If we write the view data and the business logic both inside the servlet then we are going to face 2 problem.1st is if we are changing anything on the business logic then we have to modify the view logic bz it is taking the data from the business logic.so even also we are showing the same data then also we have to modify our code**
- ✓ **The 2nd problem is to identify the business logic and the view logic is difficult to understand.**
- ✓ **So java told if you want to add some dynamic data then add inside the scriptlet inside the jsp page so that I can identify the view logic and the business logic. Servlet container put every code inside the jsp into the out.print and the code inside the scriptlet it directly put inside the servlet.**

Static factory method instantiation

When we configure a class as a bean within the spring bean configure file. then IOC-Container default try to creates the obj by using "new" operator but every class obj cannot be created by using the "new" operator and there are some classes for which instantiate the obj using static factory methods as part of the class(like Calendar class). In order to tell to the IOC-Container to create the obj by calling static factory methods as part of the class and place as a bean within the IOC-Container is called as "Static Factory Method Instantiation".

Purpose:

Static Factory method instantiation is meant for to create only one obj by the spring IOC-Container under any circumstances.

That means spring by default creates one and only one obj for any bean, i.e by default singleton but if we configure scope="prototype" then spring will creates multiple obj's for any bean. But even though a programmer make a class as singleton then also spring can creates multiple obj's using reflection api . if we configure scope="prototype" in an spring bean configuration file. In order to avoid/stop creation of multiple obj's by the spring IOC-Container and to create only one obj for a singleton bean we use Static Factory method instantiation.

Ex:

```
class CurrencyConveter {
  private CurrencyConveter instance;
  private CurrencyConveter() {
  }
  public static CurrencyConveter getInstance(){
    if(instance==null)   {
      instance=new CurrencyConveter();
    }
    return instance;
  }
}
```

```
<bean id= "currencyConveter" class="CurrencyConveter" factory-method="getInstance">
```

Whenever a class has been declared as singleton then we cannot create the obj using new operator bz constructor is private then we or IOC-Container need to call the static factory methods to instantiate the obj then such case we need to configure factory-method="getInstance" in spring bean configure file called as "Static Factory Method Instantiation".

Note:

Ex:1

```
<bean id= "currencyConveter" class="CurrencyConveter">
```

Here Spring is going to create the obj by using the reflection-api even though constructor as private and default scope is "singleton"

Ex:2

```
<bean id= "currencyConveter" class="CurrencyConveter" scope="prototype">
```

Here Spring is going to create the obj by using the reflection-api even though constructor as private and scope="prototype" i.e it creates multiple-obj's

Ex:3

<bean id= "currencyConveter" class="CurrencyConveter" factory-method="getInstance" scope="prototype">

Here Spring is going to create the obj by using static factory method and creates one and only one obj even though scope="prototype"

Instance factory method instantiation

E:\BOOK\SRIMAN SIR\Framework notes\IMP_NOTES\Spring\2 Core Advance\Material\7 Instantiation using an IFM got to this location

Factory beans

If ioc container cant create the object of some class then we need to go for factory beans. But it is the class which is meant for creating the object of the class and give to the ioc container , then ioc container place this class as beans inside the ioc container. In order to get the object by ioc container it has to call the method , so we have to create our class by implementing from FactoryBeans interface and override the methods

```
public Object getObject() throws Exception {
////this method create   the object and return the object
        }
public Class getObjectType() {
/////this method return the class type of the object
        }
        public boolean isSingleton() {
        ////this method return is the class is singleton or not
        }
```

When ever we place this factory beans class as beans in the ioc container 1st check wheather the bean id is inside the in memory data or not , if found then it pick the class name and check wheather this class is implementing from the factory beans or not , if implements then it understand it need to place the returned object of that class. Then it check the scope of the factory beans class (singleton or prototype),

Lets consider the below example

```
<bean id="reminder" class="com.fb.beans.Reminder" p:notes="Class time" p:schedule-ref="calendar" p:snooze="true"/>

<bean id="calendar" class="com.fb.beans.CalenderFactoryBeans" c:day="25" c:month="03" c:year="2019" scope="prototype"/>
```

After checking the scope of CalendarFactoryBeans it goes to the in memory meta data and check for the object(here the object is &calendar) of the beans class , if not found then it create the object of the factory beans by calling the constructor and place inside the ioc container. Then it calls the isSingleTon() method of the factory beans class , then it called the getObjectType() and getObject().

Case1------<bean id="calendar" class="com.fb.beans.CalenderFactoryBeans" c:day="25" c:month="03" c:year="2019" scope="singleton"/>

```
public boolean isSingleton() {return true}
```

Here one object for the calendar factory beans class and one object of the calendar class, bz both the class scope is singleton.

Case2------<bean id="calendar" class="com.fb.beans.CalenderFactoryBeans" c:day="25" c:month="03" c:year="2019" scope="singleton"/>

```
public boolean isSingleton() {return false}
```

Here one object for the factory beans class but 2 object of the calendar class, as the scope of the calendar class is not singleton.

Case3-------<bean id="calendar" class="com.fb.beans.CalenderFactoryBeans" c:day="25" c:month="03" c:year="2019" scope="prototype"/>

```
public boolean isSingleton() {return true}
```

Here for both factory class and calendar class we can create multiple object , if the scope of the factory class is prototype then by default the scope of the calendar class become prototype.

When to use what

If the class object has to use by other application then we have to tell the method name of our class to the other application so here go for factory beans so no need to tell the method name

But if the class is used in your application then go for SFMI or IFMI. Bz you know which method is used inside

your class

Depends on

Note

In 2 ways we can create the ioc container

- ✓ **BeanFactory factory=new XmlBeanFactory(new ClassPathResource(application-context.xml));**
- ✓ **ApplicationContext context=new ClassPathXmlApplicationContext(application-context.xml);**
- ✓ **ApplicationContext context=new FileSystemXmlApplicationContext(application-context.xml);**
- ✓ **When we are calling this line my FileSystemXmlApplicationContext class goes to the class path location read the xml file check the well form ness, check for the validity and started creating the object**
- ✓ **Then one in logical memory partition created called ioc container, then it goes to the file and read the content of the file and place as in memory meta data inside the ioc container. and then it goes to the in memory data and search for the bean defination whose scope is singleton and instantiated the object for that bean defination.**

Different between BeanFactory and ApplicationContext

- ❖ **BeanFactory is lazy instantiator(it create the object when we are requesting) and the ApplicationContext is early in instantiated as while its creating it create the object of the singleton scope classes**
- ❖ **But interviewer will tells that as it creating the object while creating ioc container it waste the jvm memory bz all the objects are not used at a time. So tell to the interviewer to write the attribute lazy=true at bean defination level , then it will not instantiate the object.**
- ❖ **But my answer is as when we use application context then when the bean defination object are creating at**

that time all the check are done(cyclic dependency check, idref check etc) so that during my running application my application is run with consistent data. Means if any wrong data will write then it will detect at the time of creating ioc container. So my application will not terminate abnormally at the middle.

But when we go for BeanFactory all the check are happened when we are requesting the object means at the run time . So it is best to use ApplicationContext.

Method Replacer

If in my application there is one class is running properly , but we want to optimize the performance by replacing the code , but if we modify the code inside the method if the new code is fail , then in reverting to previous code it cost more, bz again we have to retest my program , redeveloping cost , retesting cost, redeploy cost is high.

So we want to replace the code with the new code but don't changing in the previous code , then we have to go for method replacement .

- **Write one more class by implementing from one interface called MethodReplacer , by implementing this interface we can say that this class is replacing the method of another class , over ride one method.**
- public Object reimplement(Object target, Method method, Object[] args) throws Throwable {}
- ✓ **This class taking 3 parameter as 1st is the method name of original method for which it has to replace the logic, 2nd is the Object[] takes the parameter as which ever the original method taking as parameter, and the 3rd parameter is the object on which the original method is called.**

- ✓ **So we have to tell to the ioc container by placing the original class and the replacer class as follows**

```
<bean id="planFinder" class="com.mr.beans.PlanFinder">
    <replaced-method name="findPlans" replacer="findPlansMethodReplacer"/>
</bean>

<bean id="findPlansMethodReplacer" class="com.mr.replacer.FindPlansMethodReplacer"/>
```

- ✓ If there are 2 method are taking the different parameter means overload the method, then which method will be replace

None of the method will be replace bz both the method name is same so it cant identify which method has to replace so, it execute the original method, in this case to replace the method we have to specify in the cfg file by telling the no of parameter and the type of the parameter.

```
<bean id="planFinder" class="com.mr.beans.PlanFinder">
    <replaced-method name="findPlans" replacer="findPlansMethodReplacer">
        <arg-type>int</arg-type>    here specify the parameter type for the method which we want to replace (method i.e taking the parameter)
        <arg-type>int</arg-type>
    </replaced-method>
</bean>

<bean id="findPlansMethodReplacer" class="com.mr.replacer.FindPlansMethodReplacer"/>
```

How method replacer working internally

Generally in java we can replace one method only through override, so spring also follow the same way

- ✓ **1st whenever we are calling for the object of the class then it goes to in memory meta data and pick the bean id and search the class , perform the series of validation then it try to create the object of the bean defination**

- ✓ **As we are telling to the ioc container to replace the method so before creating the object it try to replace the logic inside the method**

- ✓ **So it 1st goes to the jvm and create one more class called PlanFinder$$Proxy by extending from the original class PlanFinder directly inside the jvm memory by taking the help of runtime bytecode generator jar(like cglib, asm jar)**

- ✓ **Then override the method findPlans and then inside this method it called the reimplement method of the replacer class, bz there only the replacing logic is written. In this way only its working.**

```
Class PlanFinder$$proxy extends PlanFinder{
    //this class is creating inside the jvm memory directly

Public List<String> findPlans(all the parameter){
  private MethodReplacer methodReplacer;
  // here it call the reimplement method
Return methodReplacer.reimplement(Object target, method, Object[] args);
    }
}
```

Internationalization

Providing the support for multiple language is called internationalization . It is of 3 level

- ✓ **Data Internationalization**
- ✓ **NLS Programming**
- ✓ **Content Internationalization**

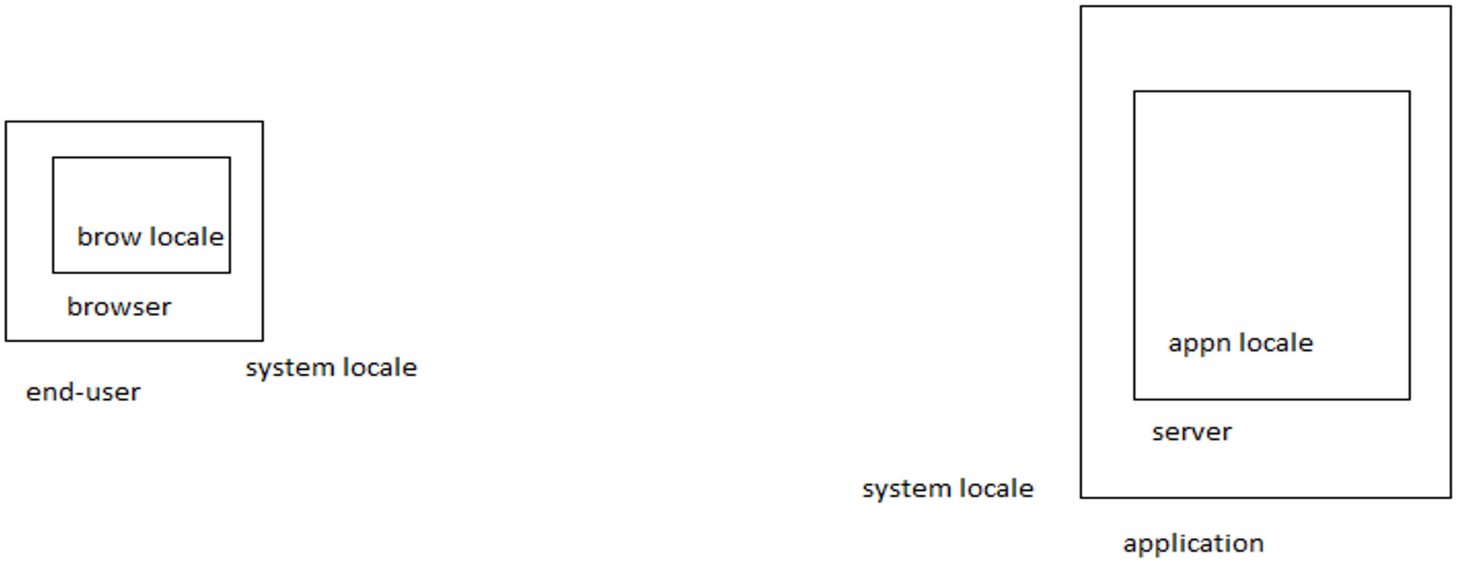
Data Internationalization----------if we want to store the data in an underline source in multiple language then it is called as Data Internationalization. At the time of creating schema we need to choose appropriate char Set for supporting multiple language.

NLS Programming---------- if you want to read or write the data in terms of multiple language while working on the application then we need to go for the NLS programming.

While working with file system we are using io stream
Character Stream----if you want to read the bits of data from the file to other place in terms of character then you need to go for character Stream (means from bits to character or from character to bits)

Binary Stream------if you don't want to convert bits to character or vice versa then you need to go for binary Stream

Content Internationalization-------



To work with showing static data of my application in multiple language we need to go for Content I18N
So to show the data in the user language we need the locale of the user(locale consist of 2 parts language code ,country code).

To get the locale of our application ----Locale.getDefault()----if we are not set the application locale it will give

the system locale, if we set then it will give application locale,
Similarly for an end user to know the locale-----request.getLocale() -----if we set server level locale then it give browser locale or vice versa.

Process for internationalization

- ✓ **Whatever the template text you want to write as part of the application on the jsp page, don't hard code the data inside the jsp page. Rather create properties file and store in terms of key value.**
- ✓ **Create one properties file for one language, and create one default properties file which you want to show as default language. And create the other properties file name append with locale , so that we can easily get the language if we have the locale,.(default is required bz if the user trying to access in different language then if the file is not there so we get exception so we need to write in the catch block the default properties file) .**
- ✓ **And while writing the properties file the key should same across all the language properties file and the value should the language .suppose we are already writing the jsp page in 2 different language we need to access the file inside the jsp page.**
- ✓ **If one file is under the WEB-INF folder then we have to read that file by taking the help of servletContext .**

```
<body>
  <header>
    <%
       String baseName="messages";
       Locale loca=request .getLocale();
       String lang=loca.toString();
       try{
            Properties props=new Properties();
            props.load(this.getClass().getClassLoader().getResourceAsStream("baseName"+lang"+.properties));
            String headerMessage=props.getMessage("key");
            out.println(headerMesage);
          }catch(FileNotFoundException e){
```

```
            props.load(this.getClass().getClassLoader().getResourceAsStream("baseName"+".properties"));
        }
    %>
  </header>
</body>
```

So we have to write this code inside the scriptlet in every jsp page , it is a boiler plate logic, so j2ee people provided one class which contain this all logic called ResourceBundle class. And every properties are called as messageBundle , with one base bundle which is the default you want to.

So the above code will be like this

```
<body>
  <header>
    <%
       String baseName="messages";
       Locale loca=request .getLocale();
       String lang=loca.toString();
     ResourceBundle bundle=new ResourceBundle("baseName",lang);
```

	String headerMessage=props.getMessage("key");
	out.println(headerMesage);

```
    %>
  </header>
</body>
```

Here is also the problem if in the application multiple messageBundle is there so we need to create multiple ResourceBundle object by passing different baseName(means if I want to display error ,message or other info in multiple language with multiple message bundle, error message bundle have error has baseName similarly the other are also having the baseName)

So we require a class to create multiple object of resource bundle and dump all the key value in one properties file so that we can get from that file.

```
Class MultiResourceBundle {
        private Properties props;

/////Constructor with parameter as props
        public static MultiResourceBundle getBundle(String[] baseNames,locale){
                props=new Properties();
                MultiResourceBundle mrb=null;
                For(String baseName : baseNames){
                        ResourceBundle rb=ResourceBundle.getBundle(baseName,locale);
                        For(String key : rb.keys()){
                        props.put(key, rb.getMessage(key));
                        }
                        Mrb=new MultiResourceBundle(props);
                }
                Return mrb;
        }
Public string getmessage(String props){
Return props.getProperty(props);
        }
}
```

So we need to change the logic in the jsp

```
<body>
```

```
  <header>
    <%
      MultiResourceBundle mlt=MultiResourceBundle.getBundle(new String[]{"messages","errors"},Locale.getDefault);
      String message=mlt.getMessage();
     Out.println("message");
    %>
  </header>
</body>
```

Here there is another problem is there if I want to read messages from multiple bundle from multi locale, then again we have to create the object with different locale, again same problem arises.
So we have to write one class which can take locale as parameter at run time.

```
Class MultiLocaleResourceBundle{

      Private static MultiLocaleResourceBundle instance;
      Private Properties props;
      Private Map<locale, Properties props> propsMap;
      Private String[] baseNames;

      Private MultiResourceBundle(){
            propsMap=new HashMap();
      }
      Public static MultiLocaleResourceBundle getInstance(String[] baseNames){
            If(instance==null){
                  Instance=new MultiResourceBundle(baseNames);
            }
            Return instance
      }
      Public  MultiLocaleResourceBundle getMessage(String key, Locale locale){
            If(propsMap. containsKey(key)==false){
                  Props=new Properties();
                  For(String baseName : baseNames){

                        ResourceBundle rb=ResourceBundle.getBundle(baseName,locale);

                        For(String key : rb.keys()){
```

```
                    props.put(key, rb.getMessage(key));
                    }
                    Mrb=new MultiResourceBundle(props);
                    propsMap.put(key,props);
                }
                Return mrb;
            }
        }
}
```

Lookup Method Injection

Question----------how servlet container works internally

Answer --------whenever we send the request from the jsp page the servlet container is receive the request ,then it reads the context root of the request identify the corresponding servlet then it check wheather the object for this servlet is already created or not if not , then it create the object of the servlet and store in the map inside the interface variable; and then it create one thread and inside the thread it calls the service method of the servlet .

Question --------how many object for one servlet is created if we send multiple request for the same servlet, is it create one object for every request or not.

Answer --------it create only one object for the servlet per one servlet defination ,(means if we declare 2 defination of the servlet inside the web.xml then it create 2 object for the same servlet, so the servlet class is not the singleton class , the defination which we are define inside the web.xml is singleton). If we send multiple request for the same servlet then it create only one object and pass the reference of the object to multiple request by creating multiple thread for each request(means if 5 request came then it create 5 thread and give the same object reference and inside each thread it call the service method of the servlet , so that all are executing simultaneously).

It create only one object bz as the lines of code inside the servlet is same for every request , if it create one object for every request then the jvm memory will be wasted. So it create one object and pass it to every one.(but in ejb environment for every request ejb container create only one object , but in web-application at one time lot of request are coming so if servlet container create one object per one request then the jvm memory will be wasted and it cant handle more no of request).

Question---------how servlet container manage multiple request by creating only one object

Answer----------whenever it gets the request 1st it check the object is created or not then , if not it create and store in the interface variable (inside the map url pattern as key and object as value). Then it create one thread for that request and inside the thread it calls the service method .

For every request it create one thread and pass the object reference to every thread, while creating the thread it create one thread stack inside the thread, in that thread stack it maintain the execution flow along with the value of the local variable of the method , parameter of the method , and the name of the person who calls that method in that line of code , so that whenever thread switching happens it pause the execution , and when it again get the chance it can execute from the state in which it left from, it didn't start from the beginning . This is how the servlet container manage multiple request by creating one object for the servlet.

Multithreading

If multiple threads are trying to access only one object, it may a singleton class object or may a non-singleton class object (state of the object) simultaneously at a time then data inconsistent will happen . To overcome this problem there are 2 solutions are there

- ✓ Declare the method as synchronized so that at a time only one thread is allowed to access that object
- ✓ Don't declare the data which is require to process the request as attribute, means at the object scope(if that data is completely shareable or read only then declare it as attribute if not don't declare) .declare it as method scope(if the

data is coming as part of the request means outside from the class then pass it as parameter, if the data is use inside the class then declare it as local variable) so that in the thread stack of the current thread the execution will be maintain so that the data inconsistent will not arise.

Question ------when to use synchronized block and synchronized method at a time

Answer -------lets take one example in a class there are 2 methods are there , and the object is access by multiple thread. While executing the 1st method or the 2nd method we want only one thread can allow to execute , means at a time only one thread can execute both the method, then make the 2 method as synchronized.

The problem is not solve by simply making the method as synchronized there may be a chance after executing the 1st method the thread switching may happen so that the 2 nd thread got the chance , and then he can modified the data inside the 1st method bz 1st thread already moves out from the method , we don't want to execute the 2 nd method with the 2nd thread modified data. So again data inconsistence happen.

To overcome this problem use synchronized block and synchronized method, put the piece of code(where you call both the method) inside the synchronized block . Then at a time only one thread can allow to call the method and execute the method

Difference Between synchronized method and synchronized block

If we make the static method as synchronized then we are achieving class level lock, but if we are making non-static method as synchronized then we are achieving the lock at the object level. But if we are making the static block as synchronized then we are achieving lock at class level always. Similarly if we make non-static method as synchronized then we are achieving lock at object level.

Scenario -------if I want to use a object across the thread , and the state of the object is not sharable, and the state of the object I want to use across the method of my class object so what to do.

Answer -------

- The first solution make the methods are synchronized and the piece of code on which you are calling this method

put inside the synchronized block. But if we do this only one thread allow to access the object , so performance issue comes.

- The second solution is put the data which the methods require to perform the operation inside the method level(if it comes from outside pass it as parameter or if it is inside the class then declare as local method) but if we do this the scope of that data is to this methods only. The other method are cant access the data.
- In such use case don't declare the data in object level or in the method level , declare it to the thread scope level only .
- So here ThreadLocal comes into picture, it's look like as map(but its not map collection) . It stores the thread level data in key and value format where the thread is the key and the value is the data. So it is not store the key it store only the value , whenever we are trying to access the data from the thread local the current thread act as key and it return and store only that data only, it return or store the data for that thread only.
- In ThreadLocal for one thread we can put only one value, it may list of value or map value as value. There are 2 methods inside the ThreadLocal i.e set() and get().
- But there is one problem with ThreadLocal , it stores the data for the Thread permanently wheather the Thread is active or inactive(after performing the operation thread destroyed). So it's the responsibility of the programmer to clean the data from the ThreadLocal before inactivating the Thread.

Look-up Method Injection

In which cases we can go for dependency injection

Target class	Dependent class	
Singleton	Singleton	we can go for Dependency Injection
Prototype	Singleton	we can go for DI
Prototype	Prototype	we can go for DI
Singleton	Prototype	we cant go for DI, bz the dependent will behave like singleton

In the 4th case we cant go for DI bz there are 2 problem is there, as the target class is singleton so one copy of

memory will be created then as the dependent is prototype means non-sharable data so , when multiple thread are trying to access the target class as it's the object level data so the data inconsistency problem will happen.

The 2nd problem is as the target class is single ton class so ioc container create only one object for that class and it performs only one time the dependency injection so even the dependent type is prototype then also it injected once into the target , and it also behave like as single ton class. We should go for Lookup-method Injection.
See the notes on K:\BOOK\SRIMAN SIR\Framework notes\Spring\2 Core Advance\Material

Event Processing

Every programming language supports 2-styles of programming

1. Synchronous Programming (Mechanism) Language

   Tightly Coupled

2. Asynchronous Programming (Mechanism) Language (Event Driven Programming)

   Loosely coupled

   Disconnected

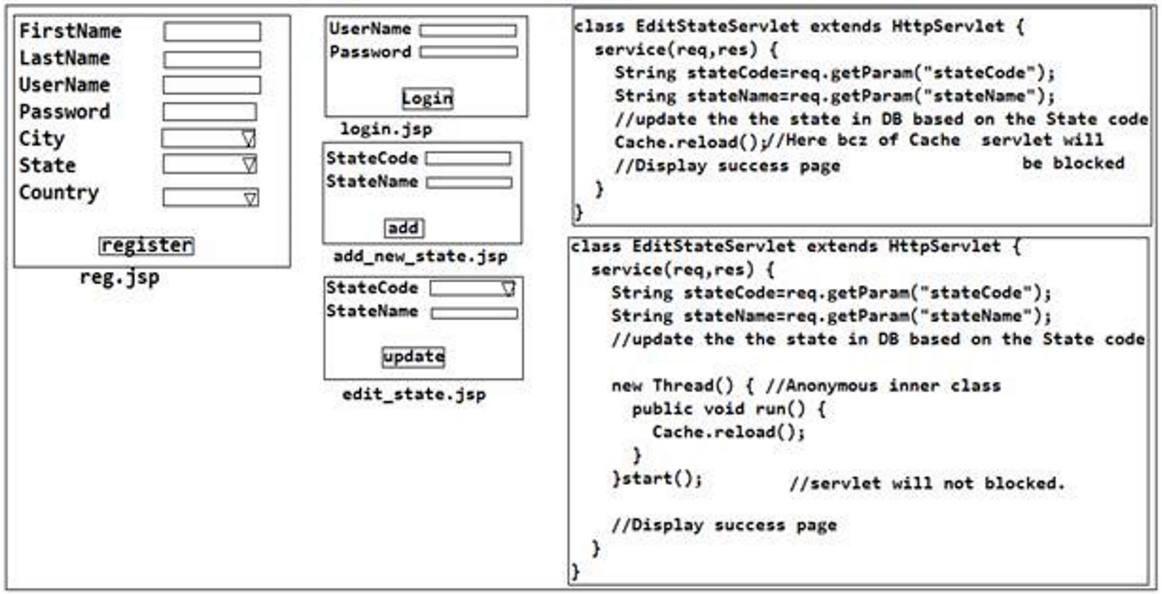
Synchronous programming is nothing but where no 2-parts of the application will execute simultaneously rather one part will be blocked by another, whenever Calle will be blocked until caller has completes the execution.
Synchronous programming is also called linear execution (one after another) of programming.

Use Case Where we need to use Asynchronous Programming:

Every Project will have Admin Module and Report Module where Application Admin (DB Admin is different form Application Admin) only allowed to change the Master table data if needed.

```
class Cache {
  private Map<String,Object> dataMap;
```

```
  private Cache() {
    dataMap=new ConcurrantHashMap<String,Object>();
  }
  public static Cache getInstance() {
    //create the Cache obj and return
  }
  public void put(String key, Object Val) {
  }
  public Object get(String key) {
  }
  public boolean containsKey(String key) {
  }
  public void reload() {
  }
}
```



If cache is reloading the data from DB then takes lot of time and leads to the time out for the user of the application hence servlet should not waits for the to complete the Cache to reload hence we need to Asynchronous prog by creating a thread and need to display the success page even though the Cache is not yet completed so that end user will feel application performance is high or quicker in response.

For example while we are doing online shopping the once the payment was success then displays success page as payment was successful after that only page reload will be happen to the particular site which is asynchronous prog other wise to reload to the site then it displays the success page the end user will fells the application is slower in response.

Cluster Environment

A. In a enterprise application the application will not be deployed in one machine, bz while the application is running more no of request is coming to the application, then as the application resource is limited then the application can not handle more no of request , so we have to go for deploying the application in multiple system. Which is called clustering the application. And each system is called as node.

B. Then each node has their ip address but their ip address will not be exposed to the world , there will be a separate system which is meant for handling the request and forward the request to the node machine. Means all the node machine are connected to that machine and that machine ip address will be exposed to the world. Which is called load balancer.

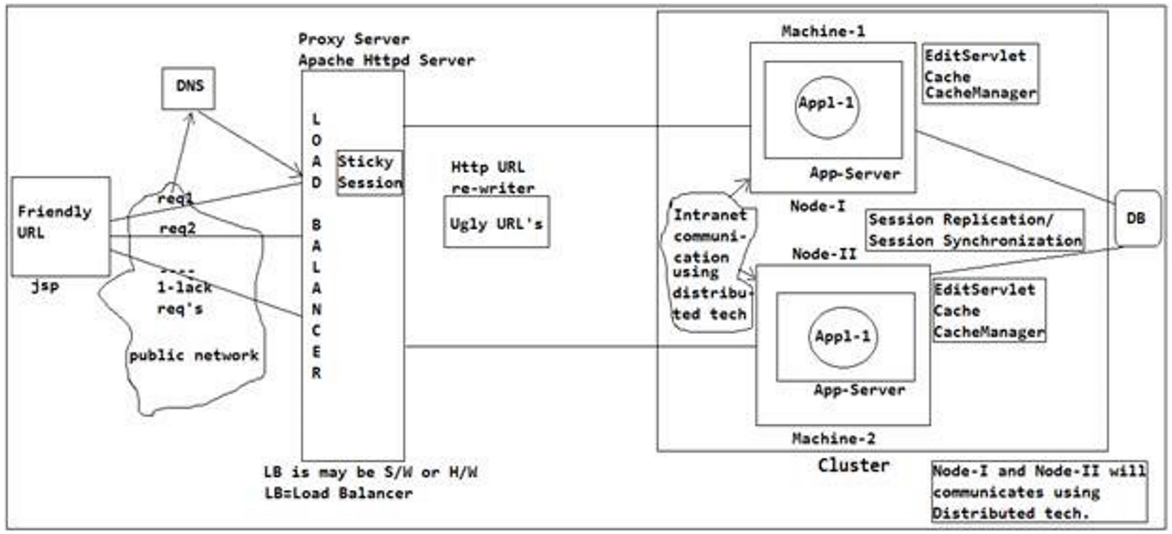
C. The user is send the request to the lbr , then lbr check which node is free so that it delegate the request to that node machine. In flipkart application user is added a product to the cart(lets take 2 node is there)Suppose it forward the request to the 1st node , then the 1st node create one session for the request and process the request and add the item to the cart, and it left then that user again came and try to add one more product to the item , now the node1 is full of request so it cant take any request more then the lbr forward the request to the 2nd node . It again create one more session for the user and added to the cart, but the previous cart will not be display bz it is added in another session.

D. So the user fail to order both product , to avoid that problem we can make the sticky session=true so that the lbr will sends the request only to the node 1 it don't forward to the node 2 as the session for that user 1st created at node1. and user can see the previous product as the request goes to the node1.but here is the problem even though the node2 is free and it can accept the request but lbr didn't forward to node2 s , the session is created at node1 , if the node1 is full of request also it wait until the node1 is free, but it didn't delegate the request to node2.

E. To avoid this problem we don't have to make sticky session, we can make the session replication so that if the session is created at 1st node it copies the session and the data to the node2 also, by serializing the data, so every

data must be implements from serializable. So if the request comes to the 2nd also the user able to see the product in the cart.

F. So lets go back to our use case when the edit state servlet trying to store the data in the cache as the cache is a class which will have 2 copy of memory in the 2 node jvm, so if the data is updated in the node1 jvm , it will not reflect in the 2 nd node, so the request which comes to the node1 will show the different data and the node2 will show the stale or old data as it is not updated . To avoid this problem distribute cache comes into picture . So in this type of case we have to go for jms.



Jms

If we developed the application1 and application-2 are developed using asynchronous distributed tech then can share the data when they are in online only that means they must be up on running otherwise not possible. But we want to send the data even though they are in offline mode then it not possible hence in order send the data even though they are in offline mode it is possible with only JMS-API using MOM-servers. That means the JMI will works based on the store and forward principle, if any data wants send over the n/w by app-1 to app-2 then JMS will stores the data in MOM-server (MOM-always up on running) with correlation id of that particular request and then sends the data to the appl-2 whenever it is in active state till that that time MOM-server will not deletes the data. This is called as Distributed asynchronous Messaging Mode communication.

JMS Domains or mode of communications

1. Point to point communication mode

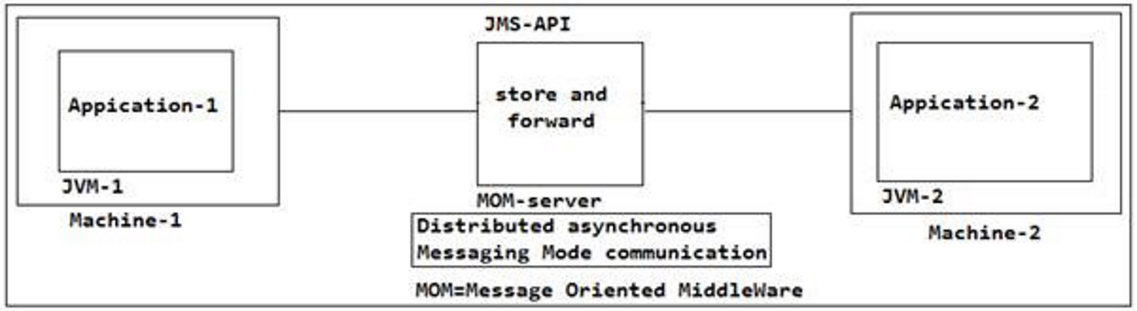
2. Publish and subscribe mode

1. Point to point communication mode:

App1 class wants to talk to App2 class then if we use JMS using Point-Point based mode then it like one to one communication. It is called queue. And there will be queue sender and queue receiver is there.

2. Publish and subscribe mode:

In this communication will happen in a broad casting manner that means message will stored in the MOM server and then all the applications or subscribers will receives the message and if any one failed to receive then the message will not be deleted in the MOM server rather it will be deleted after all the subscribers will receives the message.it is called topic data structure.



Topics are 2-types

1. Durable (Message will not be removed until all the subscribers receives so that message is guaranteed to be deliver)

Ex:

Mobile recharges top ups.

2. Non-Durable (message will be removed if anyone in offline)

Similarly Topics may be a distributed topics that means topics may not specific to specific MOM server they may even communicate with across the MOM servers if it is a distributed Durable or non-durable topic.

Cache Manager is going to refresh the Cache based on the topics that are raised in the MOM server bz it developed based on the distributed asynchronous API so that it can reload the cache.

Annotations

Problems in xml approach

1) Xml Base configuration Approach speak more / verbose . we have to remember the order of tags to work with xml . java developer has to have complete picture of xml.
2) for a java developer it's very difficult to work with xml. Bz it is not a java language
3) we can not validate declared class and package name or other information is right or wrong. When we are going for xml . there is no process we can validate our configuration is right, until we deploy the application and access that application then only we can validate declared configuration is right. XSD and DTD will validate the structure but not the data, what kind of data you have provided validated after deployment .bz of that java developer irritated while working with xml.
4) To develop an application there are several IDE's available which promote rapid application development and which make java developer life easy , but coming to xml there is not IDE's which is supporting rapid application development .
5) We have to switch from xml to java classes or vice versa to know the configuration information of the classes.

Advantage in annotation

1) Annotation speak less.(means one word only will describe whole configuration file)
2) Annotation writes along with the source code itself.
3) Annotation validation happen at the time of compilation.
4) Annotation are the java code every developer feel more comfortable while working with annotation.
5) it is every easy for developer to validate java code and provided additional information.
6) There are number of IDE's supports annotation based approach.

XML Advantages: when we configured our classes using XML:

==========

1) XML will give holistic picture of the all the classes which are available in the application.

It will give all classes information, dependencies, properties, references and more ever. By looking at XML file we can easily understand whole application and there corresponding information.

Ex:

if new developer hired into the project, He no need to visit all the classes to understand the dependencies, properties, classes and so on , by looking at Xml configuration file only he will going to understand.

2)If there are changes into classes or information into the class then there is no need to change the classes, just we can change the configuration information automatically changes are rollback to the classes. there is no need to compile the classes if there are change also.

3)If there are any bugs are available then we can easily fixed the bug by looking at xml file.

Drawbacks With Annotations:

==========

1) Unlike XML, Annotation are written under each and every class, we can not easily derive which class is depends on which other class. A particular class will describe itself rather than other dependent classes or attributes. To get holistic picture of the application we have to visit each and every class and understand the relationship between the other classes. It is every difficult to the new developer to understand the application.

2) All the annotations are written along with the source code ,If there is change in configuration of the classes then we have to change into class, if there is change into class means we have to recompile the code then only the changes gets affected. it's lead to maintenance of the application(re-compile, re-deploy, time,...).

3) if there is bug then we have to understand the complete picture and we have to visit each and every class which is related to the corresponding bug.

Question ----- what are the problem will arise by the people who are not in favor of annotation.
Answer -------the 1st problem is -----the developer will develop the application in the development environment so he

write the configuration information by keeping the view of development environment. But with the same configuration we cant deploy the application in production. We need to add some more configuration(may be the developer uses the mysql db while developing but in production the people are using commercial db like oracle rag , so the configuration will be different from development) , so deployer has to modify the configuration to deploy the application. If any bug raise during the application deploy , then both of them will point to each other , so the owner of the bug will be conflicting ----------so to overcome this problem sun micro system provide the facility that developer you write the configuration in terms of annotation and deployer you write the configuration in terms of xml file and bring both then deploy the application , the underline container will replace/ over write annotation with xml configuration, if it find any annotation similar to the xml configuration .

The 2nd problem is-while working with annotation we are writing vendor provided annotation , so that if we change the vendor our class will not work, it looses the pojofied behavior. To over come this problem sun micro system provide the retention policy . It means it develop 3 type of retention policy to work with annotation.(pojo class---- the class which can be execute by the underline jvm without taking the help of any dependency library or jars then that class is called as pojo class)

- Source-----if your annotation is under this policy then you write as part of the source code but the annotation can not be visible to the compiler or the underline jvm . So your class posses the pojofied behavior.

- Compile ----if your annotation is under this policy then you write as part of the source code but the annotation can not be visible to the underline jvm . Only the compiler can see the annotation and compile and generate the byte code . In the byte code the annotation will not be there. So your class posses the pojofied behavior.

- Runtime -----if your annotation is under this policy then you write as part of the source code the annotation can be visible to the compiler and the underline jvm .it Is there at the byte code So your class losses the pojofied behavior.bz to execute your class you need the corresponding jars.

The 3rd problem is----the xml is hierarchical so we can easily identify the dependent of the class . So sun micro system also tell you can write your annotation wherever it is required , means you can write the annotation at 4 places class level, attribute level, method level, parameter level.

Question------ you are using annotation or configuration base file to work with spring.
Answer ----------

- as part of the current project I am actually using the annotation in developing the application. Initially we have started with xml base configuration in developing the project but in the middle of the development, as the annotation seems to be becoming more popular and easy to develop the application using the annotation we migrate our self the application from xml driven configuration to the annotation at the early stages of the project.

- So initially when we are started our project development naturally we are develop with xml driven approach only , and we have develop a some quite part of the project using xml based configuration and later on after few months of the development we wanted to have our application to update the annotation , that's why we started performing the poc to understand and doing an analysis in wheather we can migrate our application to annotation or not. We spend some time in understanding it and we migrate our application completely to the annotation. And as of now I am working with annotation and I know even the xml driven approach also .

- As I have work for sometime initially on configuration , its not that this even if you look at my previous project that I have been working away back at that time that application is also built on spring and that is purely developed on xml base configuration only. and I have a good hand on experience on both annotation and as well as xml based configuration also. As we are working spring 3.5 which is completely support annotation so we migrate our application to annotation.

Question ----- as you work on both then which one you suggest
Answer -------

- **if you really ask me wheather could you recommend annotation or configuration is better, I can not go with my opinion or it is something what I feel better is not that we need to use it doesn't depends on my opinion rather there are several factors are there we need to consider in deciding to going for an xml or annotation , it does not dependent on my personal choice , I might like annotation , but I can not go for developing the application using the annotation. So I can not say go for this or go for this.**

- **On which factor you choose wheather annotation or configuration-------**

If the application that we are developing is really complicated and the no. of components with in our application are really more and the complexity in the business logic is high then rather than working with annotations I would like to develop the application using configuration only bz configuration approach will give complete or holistic picture about our application so that we can easily understand, we can maintain the application easily, debug will be easy and we can understand/derive the dependencies between the classes easily and maintainance cost will be less.

- **If the alienation is really small and the components are very less and complexity involved in the application is less then I will recommend to use annotations instead of writing lengthy configuration file so that it support rapid application development.**
- **But its not that we can use annotations only for rapid/small applications we can even use annotations for larger applications when we are properly writing project hierarchy and the naming conventions across the application (like servlet will always call business delegate and the delegate will always will call Dao like this if write) and aggregation of classes and their packages (like Dao's in dao package and servlets in servlet package) then we can easily understand without seeing the configuration file if we properly align the classes and if we organize well the project hierarchy then we easily maintain the code and we easily identify which one is calling which other component then such cases we can use annotations.**
- **That means if we properly organize the code then we can easily identify the logic without seeing the configuration but still I will not recommend to use annotations bz we cannot impose such kind convention restriction on the developers while working due to which entire application will gets wrong and may lead to confusion state.**
- **So I will recommend it is good to work with annotations if it is small application and it is better to work with configuration while working with large applications.**

How the Annotation work internally

- If we just write the annotation as part of the class , the annotation will not work , we have to say to the ioc container

that we are writing the configuration information in terms of annotation also, bz initially the support for the annotation is not there in spring ioc container.

- If we want to work with annotation ,So after creating the ioc container we have to tell the ioc container that we are writing the configuration information in terms of annotation. So there is 2 way to tell to the ioc container that add the annotation configuration to the metadata , 1st one is either using bean factory post processor or bean post processor.bz these are the only way to add some thing to the ioc container.

- So to work with annotation we have to add the post processor to the ioc container. So when ever the ioc container create the object for the bean defination it check wheather you wrote any post processor or not if you wrote then it instantiate the object of the post processor and will call the postProcessorBeforeInitialization and perform the post processing logic , then it return the object , and then ioc container perform bean life cycle on the bean defination and again call the postProcessorAfterInitialization method on the post processor here we have to write the logic for processing the annotation that you wrote.

- So to work with annotation we must add the post processor to the ioc container , for every annotation there will be the corresponding post processor class is there with the spring frame work . But to add the post processor to the ioc container we should know the corresponding post processor class name , to avoid this problem spring provide one tag Context:annotation-config . So that by seeing this tag spring will understand that we wrote the annotation , and it will add all(which ever annotation you wrote for that annotation) the post processor class to the meta data internally .

- People are telling bean factory can not support annotation this is wrong . If we are working with bean factory we have to write the corresponding post processor class as bean in the spring bean configuration file and we have to manually register the post processor to the ioc container , so that when the ioc call post processor method, the logic for annotation is written inside the method of post processor is execute and the annotation will work. So don't write the tag.

- But if we work with application context then we don't have to write the post processor inside the configuration file and register. Spring will automatically do at the time of creating ioc container . And the annotation will work.

@Required

- When we are performing the dependency injection using setter injection by default the setter injection is optional. So that to mandate the setter injection for some of the attribute we go for Dependency check, but it is removed(and also it mandate all the attribute , if you want some of the attribute then you can not) and they provide @Required annotation.

- We can write the this annotation only at the setter method level.

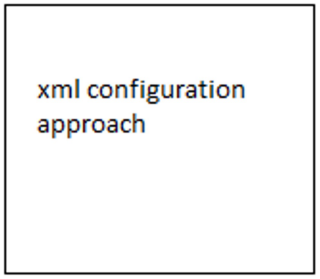
@AutoWire

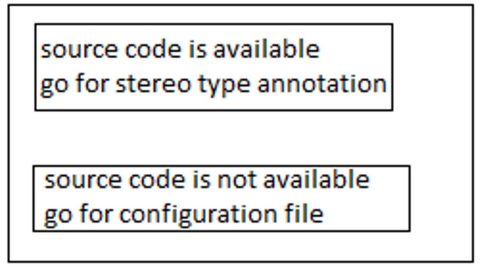
If we want to ioc container will manage the dependency then go for autowire or this annotation. We can write this annotation at 4 level constructor , attribute , method , arbitrary method level. It work only byType only, it doesn't work on the basis of byName. And it pick the injection based on where you wrote the annotation.

Contest:componentscan

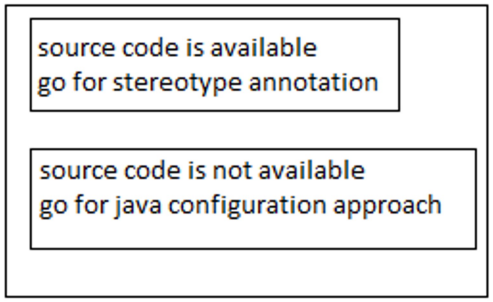
For configuration information there are actually 3 places we can configure. 1st one is write in the Configuration file, 2nd one is write annotation as part of source code , the 3rd approach is go for java configuration approach.

there are 3 combinations are there





@ComponentScan



- **In all the cases we can go for xml base approach , wheather the source code is available or not available.**
- **But to avoid xml approach we are going for annotation . But every time we can not go for annotation as we are writing the annotation as part of the source code.**
- **If the source code of the corresponding class is not there we cant go for annotation , then the only way is xml, but we don't want to write xml base approach. So go for java configuration.**
- **In this approach we are writing one class in this class we are writing all the configuration information of our application . One class is define by one method. We are annotated this class with the annotation called @Configuration**

  by defining this annotation we are telling to ioc container, the method those are annotated with @Bean , ioc call those method and this method are

  returning object , ioc place those object as bean within the ioc container , and by default the method name taken

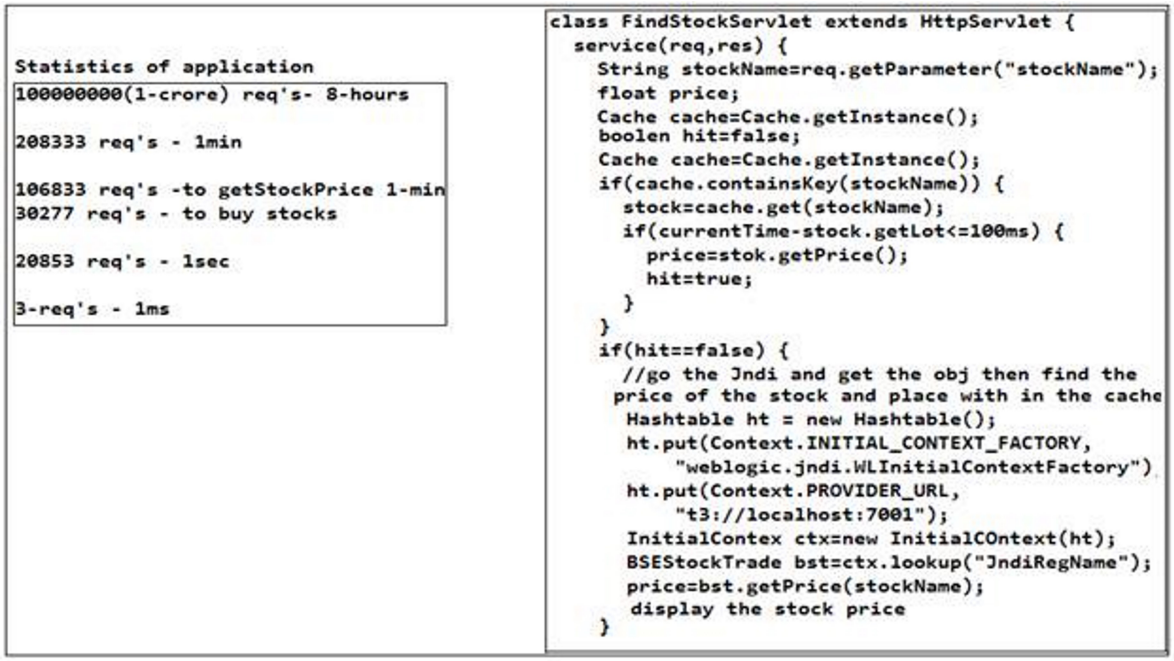
as the bean id, we can define
our own id by @Bean(name="") here we can define only one id or @Bean(name= {""}) its for giving alias name
Here we need to tell the ioc container that all configuration information are there in this class . By passing this class to the AnnotationConfigApplicationContext while creating ioc container.

- If we don't have source code for all the classes then go for either only xml or only java configuration approach, we cant go for annotation approach.
- But if we have some class are having source code and some of them don't have the source code. Then we need to go for either completely xml, or stereo Type(those who have source code) with xml(who don't have) or stereo type(who have source code) with java config(who don't have) approach.
- But in the 3rd combination as there are 2 places configuration are there, we have to tell to ioc container manually in spring 3.0 but spring 4.0 onwards spring provide one annotation called @ComponentScan

AOP

It is a new programming methodology and like procedural programming language and Object oriented programming language.

- ✓ ICICI (icicidirect.com) is one of the stock exchange web site and sharecon.com is also one of the stock exchange web site which taking the business form the BSE Stock Exchange. Due to the competition between the stock brokerage web sites to give the better performance while accessing the stock details ICICI stock web site developers though that there 1-crore customers are visiting the site so that every time going and performing the look up with BSE Stock Exchange will leads to the lot of performance issues to avoid this ICICI developers adapted the cache.



Statistics of application

100000000(1-crore) req's- 8-hours

208333 req's - 1min

106833 req's -to getStockPrice 1-min
30277 req's - to buy stocks

20853 req's - 1sec

3-req's - 1ms

```
class FindStockServlet extends HttpServlet {
  service(req,res) {
    String stockName=req.getParameter("stockName");
    float price;
    Cache cache=Cache.getInstance();
    boolen hit=false;
    Cache cache=Cache.getInstance();
    if(cache.containsKey(stockName)) {
      stock=cache.get(stockName);
      if(currentTime-stock.getLot<=100ms) {
        price=stok.getPrice();
        hit=true;
      }
    }
    if(hit==false) {
      //go the Jndi and get the obj then find the
     price of the stock and place with in the cache
      Hashtable ht = new Hashtable();
      ht.put(Context.INITIAL_CONTEXT_FACTORY,
          "weblogic.jndi.WLInitialContextFactory")
      ht.put(Context.PROVIDER_URL,
          "t3://localhost:7001");
      InitialContex ctx=new InitialCOntext(ht);
      BSEStockTrade bst=ctx.lookup("JndiRegName");
      price=bst.getPrice(stockName);
      display the stock price
    }
```

- ✓ In order to increase the performance the ICICI adapted the Cache that means for every 100-milli-sec But due to this live updates of the stock price will be gone that means taper (cause damage or make unauthorized alterations) due to this there is problem and if the BSE Stock exchange found this fraud then BSE will stops the licence of the ICICI hence now ICICI don't want to execute the Cache logic that means again ICICI wants to re-write the logic due to this maintenance problem will come.

- ✓ Hence without modifying the logic we wanted to execute primary logic and we don't want the secondary logic (Caching) we don't wanted to execute then we can manage this by writing conditional clause statements as follows.

```
class FindStockServlet extends HttpServlet {
  service(req,res) {
    String stockName=req.getParameter("stockName");
    float price;
    if(isCache) {
      Cache cache=Cache.getInstance();
      boolen hit=false;
      Cache cache=Cache.getInstance();
      if(cache.containsKey(stockName)) {
        stock=cache.get(stockName);
        if(currentTime-stock.getLot<=100ms) {
          price=stok.getPrice();
          hit=true;
        }
      }
    }
    if(isCache) {
      if(hit==false) {
        //go the Jndi and get the obj then find the
        price of the stock and place with in the cache
        Hashtable ht = new Hashtable();
        ht.put(Context.INITIAL_CONTEXT_FACTORY,
            "weblogic.jndi.WLInitialContextFactory");
        ht.put(Context.PROVIDER_URL,
            "t3://localhost:7001");
        InitialContex ctx=new InitialCOntext(ht);
        BSEStockTrade bst=ctx.lookup("JndiRegName");
        price=bst.getPrice(stockName);
        display the stock price
      }
    }
  }//service
}//class
```

- ✓ So by checking isCache is true or false we can execute only primary without executing the secondary but in order to do this we need to pass “isCache=false” as a system property at run time due to this un-necessary conditional checks will happen in the application and if this type complex conditional logic will spread across the application bcz we need caching as secondary logic in all the class of our application due to this there will be sever performance impact so avoid this problem AOP comes in to picture.
- ✓ If we write the business (primary) logic (which can work even without secondary logic) and secondary logic which can be performed on top primary logic then don’t mix primary and secondary logics in the same place rather if we

write the both the logics separately and if we give to the AOP it will combine executes both and if don't want then it executes only primary by without executing the secondary and without any conditional any checks so that performance impact will not be there which is called AOP programming.

- ✓ What are the limitations if we are going for flag driven approach----------
- ✓ when we are writing the primary business logic with the cross cutting logic using flag driven approach, then our code is surrounded with lot of conditions. So it is hard to difficult to understand the code.
- ✓ As the code is written with condition then the flow of execution is hard to understand. So that we bug fixing is going to be difficult.
- ✓ Even we don't want to execute the cross cutting logic then also it is present in our code , and unnecessarily the condition check will happen even we don't want to be execute, this killed the performance.
- ✓ As the primary and secondary logic are written intermingle with each other any change in the primary logic , should affect the secondary , so the unnecessary effort we have to put to maintain the secondary logic even we don't want to execute for only the compilation of the code.
- ✓ As there are lot of cross cutting logic are there so we have to write in all the class where ever we want, so the duplication of the code will happen.

AOP (Aspect Oriented Programming) entails breaking down program logic into distinct parts (primary and secondary) called so-called concerns.

The functions or method or logic that span multiple points (places) of an application are called **cross-cutting concerns** and these cross-cutting concerns are conceptually separate from the application's business logic. There are various common good examples of aspects like

1. Auditing
2. Declarative transactions
3. Security and
4. Caching etc.

Primary Logic:

The core logic or actual business logic of the application is called primary logic.

Secondary Logic:

Having or without having the logic the primary logic cannot be impacted such type logic is called secondary logic.

Ex: Auditing, Declarative transactions, Security, Caching etc.

Aspect Oriented Programming (AOP) compliments OOPs in the sense that it also provides modularity. But the key unit of modularity is aspect than class.

AOP breaks the program logic into distinct parts (called concerns). It is used to increase modularity by cross-cutting concerns.

The secondary logic that we are trying to write or we have to write in multiple places of application is called as cross cutting logic.

The key unit of modularity (having parts that can be connected or combined in different ways) in OOP is the class, whereas in AOP the unit of modularity is the aspect. Dependency Injection helps you decouple your application objects from each other and AOP helps you decouple cross-cutting concerns from the objects that they affect. AOP is like triggers in programming languages such as Perl, .NET, Java and others.

Spring AOP module provides interceptors to intercept an application, for example, when a method is executed, you can add extra functionality before or after the method execution.

AOP:

AOP is used to managing the cross-cutting logic to be applied on primary business logic that means instead of writing primary and secondary logics combinedly if give write primary and secondary logics separately and if give to the aop then it takes care of applying the cross cutting logic on the primary logic.

Where we can use AOP:

- ✓ Provide declarative enterprise services, especially as a replacement for EJB declarative services. The most important such service is declarative transaction management.

- ✓ Provides managing of auditing, security, logging and caching, declarative enterprise services such as declarative transaction management.
- ✓ Allow users to implement custom aspects, complementing their use of OOP with AOP.

Limitations of with OOP languages:

OOP methodology is developed by keeping view of managing the business logic as part of an application but it has not address managing of cross cutting logic is not possible in OOP languages. That's where AOP is comes in to picture.

The secondary logic will be duplicated across the classes hence we will write this in a separate class

- ✓ Here physically we seated the secondary logic and we reduced the duplication of secondary logic across the multiple places but logically we are calling the secondary logic to execute but still we are not able to avoid the secondary logic as part of application if we don't want that means we need to re-write the complete code which leads to lot of problems. That's where without referring in the method in primary logic we can execute the primary logic and secondary logic both by using AOP.

Benefits/Advantages of AOP:

- ❖ If we write the cross cutting logic in all the places of our application where ever we want then this cross cutting logic will be duplicated across all the places of our application so we need to separate the cross cutting logic from the primary business logic and even though we separate the cross cutting logic in our application and we can call where ever we want but if we call in OOP this cross cutting logic in all the places of our application then in future if we don't want the secondary logic as part of application then we need to modify the code so it is difficult hence if we use AOP then it will takes care of managing the cross cutting logic so that our application we will independent of our primary. That means we can avoid duplication of code in all the places and we can manage the cross cutting logic easily bcz if we wanted to change the cross cutting logic then we need to change the logic in one place so that we can manage the code easily.
- ❖ If we write primary logic and secondary logic combine then programmer may concentrate more time on developing the secondary logic rather than implementing the primary requirement of the application due to this release of application will be late bcz if any problem with secondary then programmer will development time will be wasted so

if we use AOP then programmer have clear vision between the primary and secondary and they are developed separately so that developer will concentrates on only primary and once after primary logic completed then only programmer will attempt to write the secondary due to this if any problem occurs in secondary logic then there is no impact on the delivery deadlines bcz already primary core business has been completed so that they can add this secondary logic in next release if it possible. That means if we can use AOP we avoid the misalignment in the goals of development of the developer so that he can concentrate/high priority to work on primary bcz both are separated.

- The chances bug fixings will be easy bcz both the primary and secondary logics will be separated so that maintenance of the code is easy.
- If we don't secondary logic as part of our application then we can easily turn-off the secondary logic and there is no impact on the primary logic bcz both are physically separated and there is no intervention in execution of primary due to the secondary.

Principles of AOP

there are 7 principles are there ---ASPECT,ADVICE,JOINPOINTS,POINTCUTS,WEIVING,PROXY,TARGET

ASPECT

Separate the cross cutting logic which you want to apply on multiple classes from the primary logic and write it in another classes, that piece of code that you separate is called aspect. It is nothing but piece of code.

-----technical defination-----

Aspect is the piece of code that you want to advice on the join point of the target class to build the proxy.

ADVICE

As we want to execute the primary logic along with the cross cutting logic , it is separate from the primary logic , so we have to tell to the aop how we want to execute that piece of code. This is called advice. There are 4 way we can execute secondary logic.

Before Advice-----before primary logic we want the aspect to be execute.

After Advice-----after primary logic we want the aspect to be execute.

Around Advice-----before primary logic we want the aspect to be execute, and after primary logic execution also we want the aspect to be execute.

Throws Advice-----when any exception will arise at the time of primary logic execution.

-----technical defination-----

Advice is the way we want the aspect to apply on the join points of the target class to build the proxy.

JOIN POINTS

It is the place where we can apply the cross cutting logic of our application, means we can apply anything on the class at 5 places , i.e method, constructor, static block, instance block, finalize method or destructor. So there are 5 possible join points are there. Out of which spring support only one join points i.e method.

-----technical defination-----

Join points are the places on the target class where we can advice our aspect .

POINTCUTS

These are the collection of places where we can apply our cross cutting logic. Means if a class contain 5 methods , then we have 5 join points and collection of 5 join points are called pointcuts.

-----technical defination-----

Point cuts is the collection of join points in the target class.

WEAVING

It is the process of combining our cross cutting logic with the primary logic to execute both the logic together.

-----technical defination-----

Weaving is the process of combining the advice aspect and the join points of the target class Together .

TARGET

It is nothing but the class where we want to apply our cross cutting logic.

-----technical defination-----

Target is the class where we want to advice our aspect to build the proxy.

PROXY

It is the result of combining the primary logic with the cross cutting logic.

-----technical defination-----

Proxy is the result of Weaving , means the out put of Weaving process.

Point cuts

Actually there are 2 point cuts are there Static and Dynamic point cuts.

Static----- static means the method on which we are going to advice our aspect are fixed at the development time, means whatever the input we are passing always these methods are acting as join points.

Dynamic ------Dynamic means the method on which we are going to advice our aspect are fixed at the runtime based on our input. These join points are choose at the runtime.

Weaving

There are 2 types of weaving are there compile time and runtime weaving.

Compile time Weaving-----in this process 1st we have to compile our target class and Aspect class using java compiler, after compilation the generated .class file , we have to pass it to the Weaving compiler. Along with .class file even the configuration file also.

Then it generate another .class file which is called as proxy.class file. As it is a 2time compilation process it kills the development time, if any change we have done then we have to again compile. Also after changing in the target class if we forgot to run weaving compiler then we are using actually the old proxy class , but we are thinking new proxy, this create confusion.

Runtime Weaving-----in this process one component called runtime weaving is present inside the application jvm internally. After loading the byte code of the target class and aspect class , when we are requesting the object for the target class , from runtime weaving , then it will check wheather on this class is there are some aspect to be advice or not , if not it returns the reference of same target class object.

If it finds that on this class , there are some aspect to be advised then it combine the target class and the aspect class and create one new object called proxy class at run time, and returns the reference of the proxy class.

Question = = ===what is the result in combination of aspectj and spring aop=====

	Spring support	**Aspectj Support**	**Result**
Join points	Method	Constructor, instance block static block, Method Finalize method	Method
Point cuts-static ,dynamic pointcut	Dynamic	Static point cut	Static point cuts

Weaving -compile, runtime Dynamic Weaving Compile Time Dynamic Weaving

Around Advice

Lets think that we are performing the logging cross cutting logic for my target class. So I want to track the execution flow of my method . So I want to perform the aspect before and after calling my method also. So we have to go for Around Advice. Around advice is something that we want to perform before and after my method execution.

- Lets think that we already created the proxy object by combining the aspect and the target class join points. So we are calling the target class method on the proxy object. So what happen lets discuss.
- As it is around advice when we are calling target class method the control instead of target class method goes to my aspect advice class invoke() method . To perform the logging operation on the target class method we need complete information about the method for which the invoke() has been called. So methodInvocation comes into picture.
- We can get the method name by calling methodInvocation.getMethod().getName(); we can get the parameter of the target class method by Object[] args=methodInvocation.getArguments(); .
- Then we can perform the logging operation and as instead of target method invoke() method called so it is the responsibility to forward the control to target class method by calling one method called proceed() .
- Then after execution of target class method we can perform again the logging operation as we only get the result from the target class and we are returning the return value to the proxy class object , not the target class.
- What are the controller is within the aspect class when we go for around advice----------
- As instead of target class control will comes to invoke() , we can see the information about the target class method

using methodInvocation. As we can see the parameter we can even change the parameter also. And as we only means invoke() is calling the target class method we can call the target class method by the modified value also.

- As the control comes to invoke() , and we only are calling the target class method , so we can control the execution of the target class method, means if we don't call proceed() the control will never goes to the target class method, its upon to us wheather we can call or we don't call its our choice , if we call target will execute else it will never execute.
- As we only are calling means inside invoke() we call proceed() so whatever the output will come from the target class method will comes to invoke() the proceed() statement only, bz here only we are calling the target class method, as the result is on our hand we can even modify the result also and we can return the modified value also to the proxy object.
- For further internal flow see the example of around advice.

Configuration based AOP approach

Whenever we create ioc container using ApplicationContext then it reads all the meta data and create in memory logical partition and load the meta data, then it search wheather you wrote any bean factory post processor or bean post processor, if wrote it register those processor by instantiating and place in the ioc container. Then it looks wheather you wrote any aop configuration or not . If wrote then it doesn't instantiate the original object it advice the aspect on the target class and then it create the proxy class object and store in the ioc container.

Pointcut Designator's (PCD's):

What are Pointcut Designator's (PCD's) and How many PCD's are there in AOP?

There are several pointcut designator's (PCD's) are there that permits us to advice our aspect at different levels with in a class.

1. execution()
2. within()
3. this()
4. target()

These 4-are coming from AspectJ

5. beans()

beans() is coming from spring.

1. execution():

It always talks about what can be executable that means methods.

Syntax:

execution(accessModifier returnType package.className.method.(no.of params type))

accessModifier is optional to place but if wanted to place then we should not place * rather we need to place public, private etc. If we don't want then don't place accessModifier

Ex:

execution(* com.mba.beans.LoanApprover.*(..))" here accessModifier we didn't specified so by default takes as returnType.

Ex:

execution(public boolean com.mba.beans.LoanApprover.*(..))"

execution(boolean com.mba.beans.LoanApprover.*(..))"

execution(* com.mba.beans.LoanApprover.*(..))"

2. within():

For example we wanted apply the cross cutting logic for all the service layer or transnationality layer of our application then within() designates classes that are there within a package.

within(com.mba.beans.*(..))"

within(com.mba.beans.LoanApprover(..))"

We can achieve this by using execution() as well but if we use within() then the expression gets evaluated time will be less bcz basically within() meant for to designate the classes that are there in package so that it can quickly generates the proxy by overriding the required set of methods of specified classes.

Cost of advising/time required to advice is less that means proxy will be created much quicker.

Differences between the execution() and within():

execution() - for matching method execution join points, this is the primary pointcut designator you will use when working with Spring AOP

within() - limits matching to join points within certain types (simply the execution of a method declared within a matching type when using Spring AOP)

In other words, execution matches a method and within matches a type.

So by method means execution() will check by method that 1st it directly picks up the method then which class then which package then it will applies the advice so time will be more to evaluate the expression but within() takes less time to evaluate the expression bcz it starts evaluating the expression using by type so it directly goes to the classes then in that classes which are advised.

Differences between this and target:

this - limits matching to join points (the execution of methods when using Spring AOP) where the bean reference (Spring AOP proxy) is an instance of the given type

eg: this(com.xyz.service.AccountService)

any join point (method execution only in Spring AOP) where the proxy implements the AccountService interface

target - limits matching to join points (the execution of methods when using Spring AOP) where the target object (application object being proxied) is an instance of the given type

eg: target(com.xyz.service.AccountService)

any join point (method execution only in Spring AOP) where the target object implements the AccountService interface

3. target()

It talks about to specific target class and searches the pointcut by target class type but not the proxy class type. Bcz target class can have sub class as well then we can assign sub class reference to the super class so that it will performs the pointcut for both the classes as well.

Ex:

target(com.aa.beans.Calculator)

4. this()

It talks and searches for the pointcut based on the specific proxy class obj.

Ex:

this(com.aa.beans.Calculator)

5. bean()

Spring 2.5 features a new pointcut designator – bean() that allows selecting join points in beans with a matching name pattern. Now it is possible to use the auto-proxy mechanism along with Spring-AspectJ integration to select a specific bean even when there are more than one beans of a type.

Besides selecting a specific bean, this pointcut designator offers two interesting ways to select beans if you follow an appropriate naming convention:

<bean id="calculator1" class="Calculator">

<bean id="calculator2" class="Calculator">

<bean id="loggingAspect" class="LoggingAspect">

Summary:

The different PCD's are taking about different aspects of our classes that has to be applied and these different PCD's are given to enhance performance in creating the proxy.

execution- talks about method

within- talks about package (within in this packages certain classes)

target- talks about class

this- talks about Proxy class or Proxy class obj.

beans- talks about beans in spring bean config file and we have option where we can write bean(*service) etc.

Spring JDBC

Classic Approach---------

- When we are going for the Spring jdbc we need to setup some basic class configuration. Which is , incase of jdbc if we want to perform some operation then we need Connection, and the other thing, but when we are go for spring jdbc , we don't need to create the connection , statement all the boiler plate logic is handle by the jdbcTemplate.
- But to perform the operation the jdbcTemplate need the connection , to create the connection we need to give the information about the underline database. So we need to configure all the data base configuration in a data source, then from the data source jdbcTemplate will take the information and create the connection and perform the operation.
- Classic Approach works on two phase i.e creator phase and the callback phase. And it is works on totally call back mechanism. Means lets take two class as example , one is dao class and the other is jdbcTemplate class.
- Dao class want to perform the operation , and it has the code to perform the operation, but to perform the operation it need the connection and prepared statement object as an input. But it doesn't have the input. The input is with the

jdbcTemplate . jdbcTemplate does not give the input to the dao class.

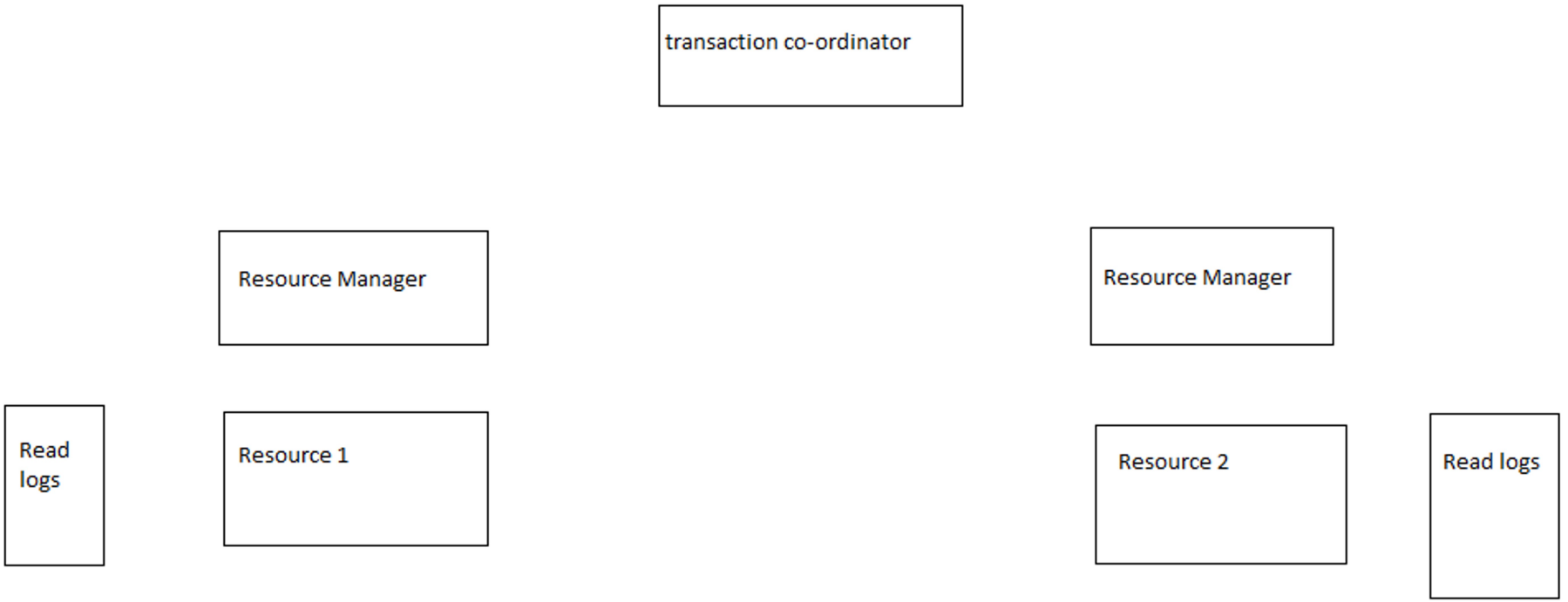
- So the jdbcTemplate has to call the method on the dao logic by passing the input as parameter and perform the operation and give the out put to the dao class . This is called as the call back mechanism.

Spring Transaction

Transaction means performing multiple work as one unit or at one shot is called transaction. It is of 2 type local and global transaction.

Local Transaction-------within the transactional boundary if we are performing the operation on one transactional resource. Then that transaction is called as local transaction.

Global Transaction-------within the transactional boundary if we are performing the operation on multiple transactional resource then that transaction is called as Global Transaction.



- When ever we are working with global transaction we cant go to the Resources asking for connection or the other thing , we have to go for one component called Transactional co-ordinator . Which is provided by every application server.
- Whenever we are going to the transactional co-ordinator asking for the connection first time then it will generate one transaction token and give to the application, and then it goes to each and every Resource Manager and give this token and tell prepare for the operation. Then this Resource Manager will goes to the corresponding Resource and tell prepare for the operation.
- After performing the operation the data will not be persisted in the database (lets take one example as data base) it will be store as part of the session of the data base for this transaction token. And we are not going to commit or rollback directly on the Resource , we have to do it on the transactional co-ordinator by passing the transactional token.

- Then the transactional co-ordinator generate one prepare token and send it to the Resource manager and ask for preparing for commit or rollback. Then manager pass the token to the Resource , then the resource will perform some validation wheather the data for this transaction which is store as part of the session is compatible with the existing data or not (means if the data store the database will work or throw some exception) . If it success then the Resource will write this change information one read logs and lock the data base for that transaction. And send the msg to manager , that I am ready. Manager will send back it to the co-ordinator.
- Then co-ordinator does this job with all the Resource and finally when all are ready then It sends one commit or rollback token to the Resource one after another.
- Case1----if all are successful then data will be stored and execute successfully.
- Case2 ----if while doing the compatible operation if any resource is fail then it sends the rollback token to all the previous resource and current also . And all are going to be roll back.
- Case3----- if while commit one of the Resource doesn't perform committing or rollback(if the database is crashed) Then the co-ordinator wait for response from the resource to a certain amount of time , which is called transactional time out . And after it sends to all the previous resource which are already committed or roll back , one rollback token and say rollback. But if the resource is already committed how to rollback,
- So the resource do magic here , as it recorded whatever it perform on himself in the read logs , it identify and delete I tone after another , that's how the Global Transaction works internally.

Question -----why we generally goes for multiple database

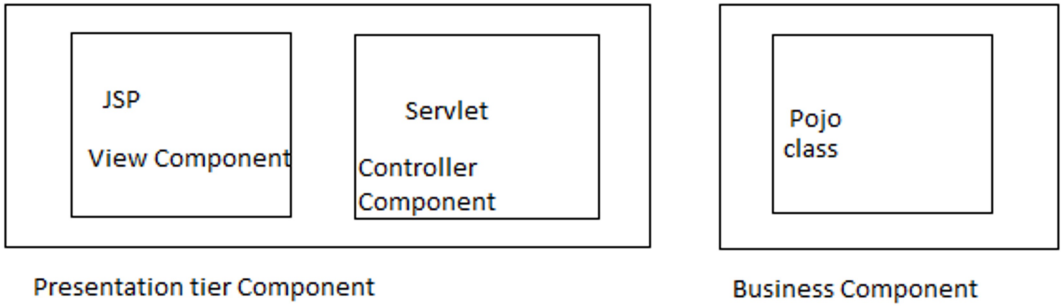
Answer-----example1----as the auditing is the common aspect for one application, and we are storing the audit data in the database, and if we store both in the one database , then while performing the data base backup we have to shut the application down, then only we are going to perform the backup in the database. And as both data are store in one database it takes more no of time , so the business is going down . So it is recommended to store the auditing data in different database.

Example2----in the company generally we are maintaining 2 database, one is for storing employee and one for salary database. Bz we cant store ,Bz we cant store the confidential data and the general data in one database. So while performing the salary data update operation we have to update in both the data base so we have to go for Global

transaction.

Project Architecture

Actually the architecture is based on the technology which you choose lets take we are building our application using the servlet and the jsp technology.



- When we are choosing Servlet and jsp as presentation tier logic , we have to follow some rules in building the application. Here JSP and The Servlet is called as Presentation Tier component. And the pojo class is called as the business Component. Using the presentation tier component only the user can communicate with the application. He can submit th req or can get the response using the presentation tier component.
- When the user submit the req we can handle the request by writing the business logic as part of the jsp page .but it is not recommended to write the business logic as part of the jsp page Inside the scriptlet.
- JSP is called as the view component bz all the data displaying page or data rendering logic are written as part of the jsp page. We shouldn't write the business logic as part of the jsp page.
- When the jsp page forward the request don't let the req received by any other jsp page or any component, let the Servlet receive the request. It is recommended that the req is received by the Servlet.
- Here Servlet is called as the Controller Component. Bz it is the ultimate decision maker of the request. What to do to

process the request, or after processing to whom to delegate the response, or whom to delegate the request to process, what method has to be called for the req, what page to shown after processing the request, all such decision is in the hands of the Servlet, so it is called as the Controller Component.

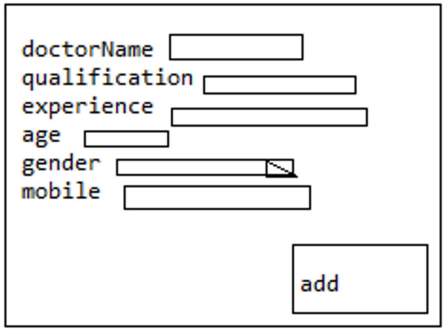
- Don’t write the business logic as part of the Servlet , rather write it in a separate pojo class , and call the method on the pojo class and pass the required information to process the request. Here Servlet only has to read the request parameter and call the method on the pojo class by passing the data.
- Why we should not write the business logic as part of the Servlet--------
- If we write the business logic as part of the Servlet , if we want the same logic to be used in another component of my application then we cant use. Let us take one example in our organisation we have one web application , and inside the organisation we have one more desktop application . Both the application are need the same business logic . So if we write the business logic as part of the Servlet then the desktop application component cant use business logic . Bz servlet is the technology specific component.
- But if we write the logic in a pojo class , in servlet we can call the class method, and for the desktop application we can make the pojo class as jar and place as part of the desktop application bz it is not technology specific.
- If we write the business logic as part of the servlet , then it is going to be tightly coupled with the technology specific presentation tier component , if we change the presentation tier technology then we have to redevelop the whole application from the scratch.
- If we are using Servlet and Jsp as technology then if we write business logic in servlet then if we change to strut or spring then we have to modify whole application .
- If we write the business logic as part of the pojo class then how the class will be look like. Don’t pass the data from the servlet to the pojo class method as parameter , rather declare all the input data as parameter , and whatever it returns also as part of the attribute. Allow the method to use the data by using the state of the object, throw encapsulation or abstraction.
- As the state of the pojo class object is non shareable, thread safe issue will be arise. So create per request one object of the pojo class , then interviewer will ask why we should write the business logic and the persistence logic as part of the pojo class.
- So we have to tell as we are choosing the Servlet and the jsp as Presentation tier technology so it’s a small or

moderate application , which has not much amount of complex business logic or complicated persistence logic. If you are thinking the business logic is much complicated then I recommended to you that you should change the presentation tier technology also. Bz using servlet and jsp we cant build enterprise application. As it is small application that's why we can write the business and persistence logic as part of the pojo class itself.

- If you are insisting me to write them separate you can write but I will suggest that don't write them separately.

Problems in servlet jsp

When we want to add something to the database we need to read the value from the jsp page and bind to the doctor object and we need to pass the object to the business logic class. So if we need to add a patient we need to copy this logic in addPatientServlet also so , write a RequestHelper class and write the below logic, instead of we write this logic go for framework.



```
public class AddDoctorServlet extends HttpServlet{
       RequestHelper helper=new RequestHelper();
       Doctor doctor=helper.wrap(HttpServletRequest ,Doctor.class);
      //pass this doctor object as an parameter to the business class
as input
}
```

```
class Doctor{
       String doctorName;
       String experience;
       //follow the naming convention in writing this class
that the request parameter name that is passing from the jsp page
should same as the attribut name of this class so that we dont need to
write the boiler plate logic
}
```

```
public class RequestHelper {

public  Object wrap(HttpServletRequest req,ClassType<> classType) {

       Map<String,String> parameters=new HashMap();
       parameters=req.getParameters();
       Object obj=classType.newInstance();

       for(String paramName : parameters.keySet()) {
             String paramValue=parameters.get(paramName);
              Method method=classType.getDeclaredMethods("set"+paramName);
              obj=method.invoke(obj, paramValue);
              }
              return obj;
          }

      }
```

Similarly we have to write all the validation logic also, instead of we write go for framework

```
public class AddDoctorServlet {

	boolean flag=false;
	List<> errors=new ArrayList();

	try {
		int iAge=req.getParameter("age");
	}catch(NumberFormatException e) {
		flag=true;
		errors.add("the age should be integer");
	}
	try {
		int iExperience=req.getParameter("experience");
	}catch(NumberFormatException e) {
		flag=true;
		errors.add("the experience should be integer");
	}
	try {
		if(mobile.trim().lenget>0 && mobile.trim().length()>=10) {
			int iMobile=req.getParameter("mobile");
		}else {
			throw new NumberFprmatException();
		}
	}catch(NumberFormatException e) {
		flag=true;
		errors.add("the mobile should be 10 digit no");
	}
	if(flag==true) {
		req.setAttribute("error-msg","errors");
		read all the value set to the request scope
		req.getRequestDispatcher(/addDoctor.jsp).forward(req,resp);
	}else {
		Doctor doctor=helper.wrap(HttpServletRequest,Doctor.class);
		pass the doctor object to the business class
	}
}
```

So to handle all the above problem we need to go for one architecture, using which we can implements best coding

Architecture

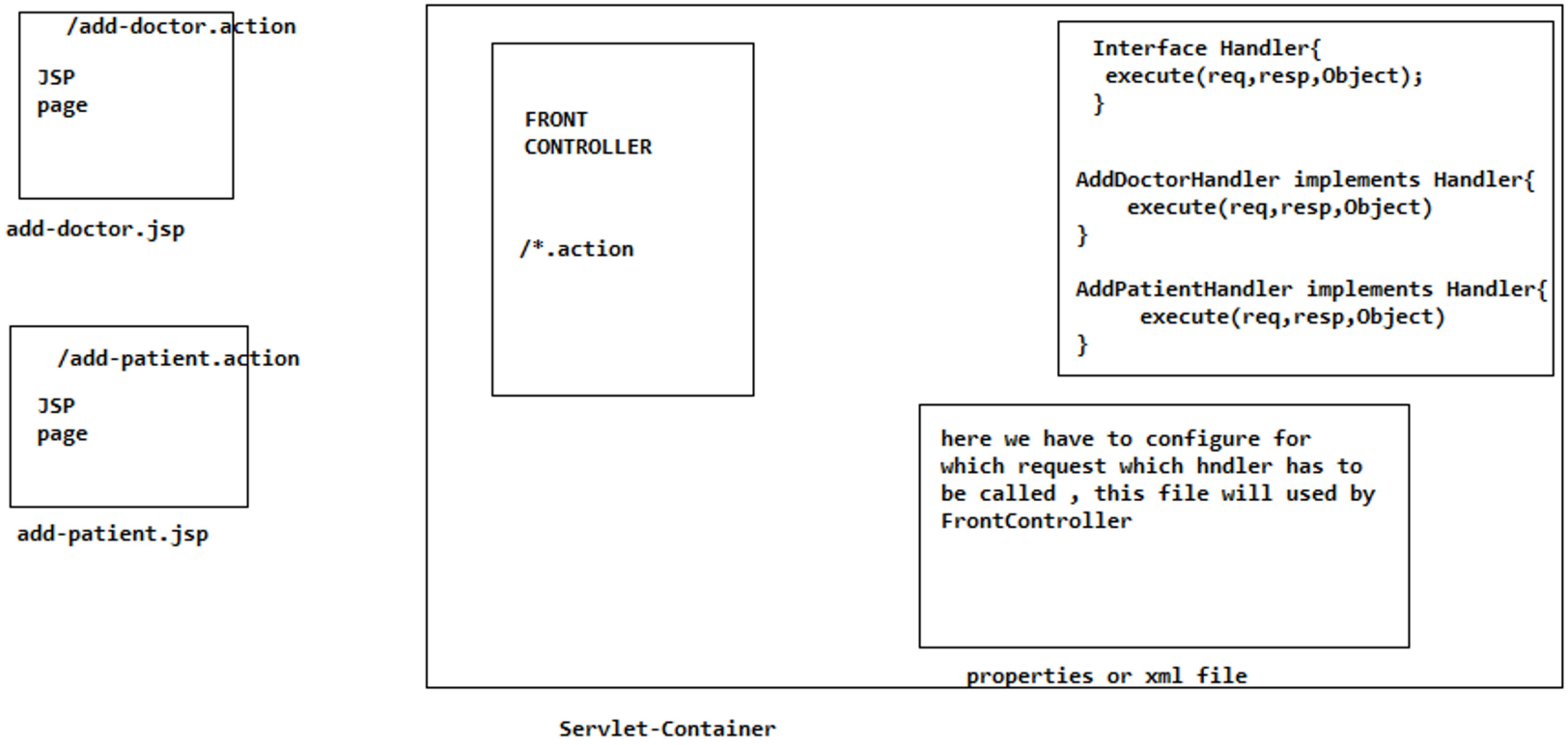
- If we are considering above logic , that for every request we have to perform request wrapping and data validation logic, so in every class we have to call the method on the RequestHelper , we have to create the object of request helper class , so this is something is the boiler plate logic .that we have to write in every servlet within my

application.

- So don't write the common logic that you want to apply for all the request that are coming to my application. Write it in one class and let the class perform this common logic and after call your servlet. So FrontController comes into picture. Let all the request will receive by FrontController and FrontController will call your servlet.
- Purpose of FrontController-----write your all common System Service logic inside the FrontController so that it is become the single entry for your application and it will handle all the common logic for all the request that are coming for the application.
- The second purpose of the FrontController is it is become the central entry of the application so that it will filter all the request means , If any invalid request comes it is received by FrontController and it not allow the request to entry into the application.
- The third purpose is if we want to apply any additional functionality like auditing, logging in all the component of your application we can write it inside the FrontController.
- So you are telling that every request which is coming to application will receive by the FrontController , but to receive request it has to call by Servlet Container , bz when request comes from the jsp page it, will receive by the Container then container will identify and forward the request to corresponding Servlet.
- So if for all the request FrontController will receive the request then we have to configure it inside the web.xml with all the url pattern that are possible to come for the application. But if we configure all the url pattern for the FrontController maintainability will increase. The other way is we can configure it by /* as url pattern so that it match for all the request url pattern. But if we do this then irrespective of valid or invalid request all are receive by the FrontController.
- But we can follow one pattern we can configure the Controller by /*.action , and we have to configure all the servlet with url pattern urlpattern.action so that for all the request FrontController will receive the request. So finally for all the request Controller call FrontController .
- FrontController call all the servlet of my application. As all the classes are called by the FrontController then the classes don't need to be a Servlet classes. Bz those are not called by the ServletContainer. So in the application there will be one servlet class is there. That is FrontController.
- So all the classes will have same functionality that it receive request, response and the object as an input so that it

pass the object to the business class as an input , and perform the operation and return the response. this classes are called as handler class and those are implementing one common interface. But this classes has to call by FrontController then for which request which class has to called FrontController don't know. We have to tell to the FrontController for which request which classes has to be called. We have to write it in a properties or xml file , so that after performing all the system service for the incoming request it goes to the file and identify the classes and forward the request with input.

- But all this logic we have to write , instead of we write this logic go for framework



- When ever we submit the request from the jsp page the request comes to the servlet container. In case of struts we have to configure the action servlet as *.do , so the request will be received by the ActionServlet. Here the ActionServlet acts as Controller Component.

- All the common system services will be done by the Action Servlet , then for further processing the request it has to call the action class, but it don't know for the request which action class has to call, so we have to tell to the Action servlet by passing the struts-config.xml .
- As part of the file we have to configure the ActionForm class also, it is the class in which the data has to be binded. In the action form class we can configure validation logic , when the request wrapping completed then the ActionServlet go to the xml file and pick the corresponding action class for that request. And pass the req, response, action form class and the action mapping as input.
- Here the action form is the object in which we binded the data for which the request is coming, and we binded the action class to the action mapping, after processing the request where to pass the control we are configuring as part of the struts xml file . So it can go to the correspond action mapping and get the information where to forward the request that's why we are passing action mapping as parameter.
- As we are returning from where to forward the request , so the return type of the method is Action forward.

Struts Project Architecture

- Here jsp acts as an view component of the application, whenever the request coming from the jsp page , my Action class will receive the request(here we are not discussing the framework architecture we can say req received by Action class) can we call Action class as the Controller component of my application?
- We can not call the Action class as Controller component , bz the decision making over the req is not there with the Action class , the decision is already taken by the ActionServlet , here the Action class only acts as a Command

class, bz it has the fixed logic to perform . So we can not call this class as Controller Component.

- So write the business logic in Business delegate class and the persistence logic in the Dao class . It stands for Data Access Object. All the persistence specific logic don't write as part of the other class , write It in the Dao class so that if any change happen as part of persistence tier it will not impact the other part of the application.
- There are lot of benefit we can get when we separate persistence logic from the business logic, if we change one database vendor to another database vendor , we don't need to change the business code , we only need to modify the Dao code, but if we wrote both as one then we have to modify the Dao along with the business logic also.
- If we change one persistence tier to another then the current code will not work so we have to modify the dao logic, but if we write both as one then we have t modify both. If the way which we are storing the data as part of the database is change means the table name or the column change then we have to modify the dao logic also, but if write both as one then we have to modify both.

Encapsulation And Abstraction

In our application we are using dao class for the persistence operation , so that we are abstracting all component of our application from the persistence tier. If any change in persistence tier that is not affecting the other component of our application. Here we are abstracting the persistence tier from the other part . here we are using abstraction.

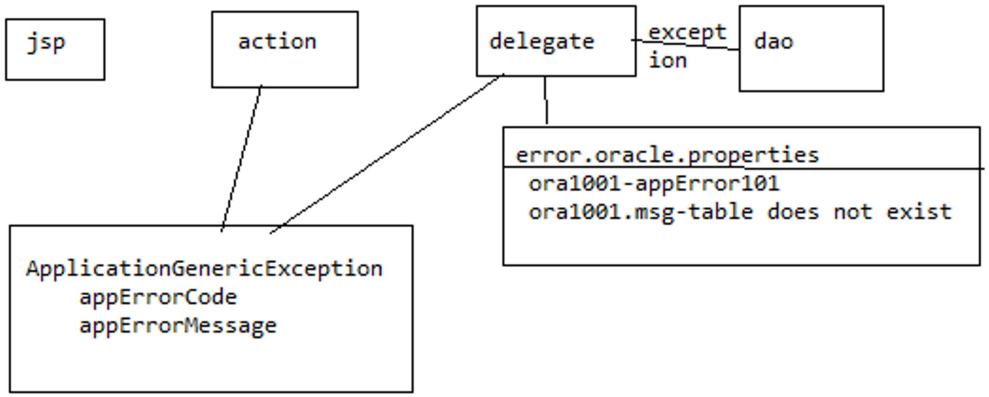
We are accessing the dao class through the method only we are not accessing directly the dao logic. So that any change in the dao we are not going to modify bz we are accessing the method . Inside the method what is change we don't know. So we are encapsulating the complex logic of the dao class from the business delegate by giving the accessibility through the method.

- Why we write the business delegate class ----people think we write the business logic as part of the delegate, but its not correct ,so what will we write as part of the delegate lets try. Lets take delegate is not there , so when the action class forward the request dao will receive the request, but the dao not always perform well, sometimes it fail in processing, if he fail then dao has to report back the error by throwing the exception.
- When it throw the exception, it comes to the action class as action call the dao, so action has to tell to the user for reason of failing, it should not return dummy value or some generalized exception back to the user. But to tell the reason for the exception the action class has to read the error code from the exception.
- Lets take we are using oracle then action has to read the oracle specific error code i.e ora1,ora2 like this , then if we

change from oracle to mysql then the logic for returning the oracle error code will not work in mysql. Inside the action class. So we have to modify the action class, then again the problem is arise that if any change in the persistence tier will impact the other part, how to overcome from this problem. Actually we can solve this problem in 2 way for technical application or for business application.

Technical Application Error Handle

- lets have the business delegate back, so action will call the delegate and the delegate will call the dao, so when the exception occur dao should throw the exception , so delegate receive the exception. Upon receiving the exception it should not throw the technology specific or persistence tier exception. It should use his own generic exception class for reporting the exception .
- Here In the exception class we are storing the appErrorCode and the appErrorMsg as attribute in which we are populating the data from the error.oracle.properties file(if we are using oracle as database, if some other then some other file).we have to identify all such possible exception and what message we have to show to the user , track all such possibility and place them as part of the properties file .
- So whenever we are receiving the exception in the business delegate then translate the technology specific exception or the persistence tier specific exception to appropriate business exception by taking the error code from the exception go and check in the properties file what Is the application specific error code which you configure take the appErrorCode and take the message and substitute in the generic exception class and throw the generic exception to the action class.
- So that the action class convert the exception to an user friendly exception, and throw to the user so user feel comfortable by seeing the exception. We have benefit here that the action class don't know from which persistence tier the exception is coming , if any change in persistence tier we have to change in the properties file ,not in the action class , my action class don't need to know about the persistence tier.



Business Application Error Handle

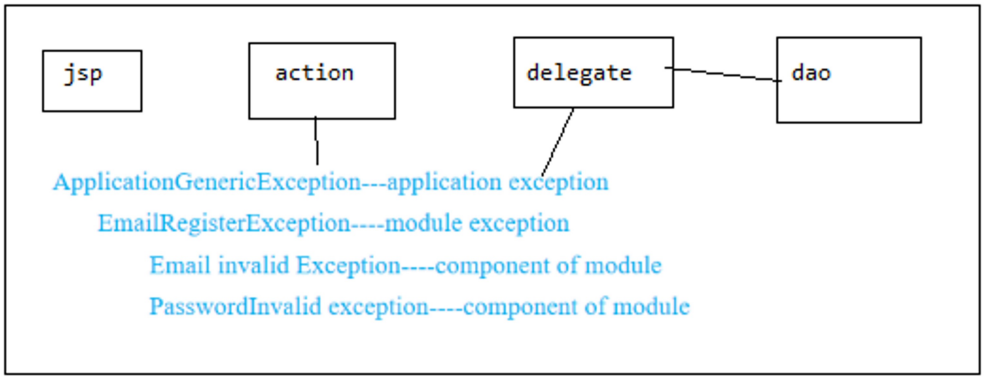
- We can not implement the above error handle logic as part of the business application we have to follow some other methodology . When it comes to business application 1st create one application generic exception , and then create exception in the hierarchy means divide the exception module wise.
- For one module create one exception class , and then for the component of the application create one one exception , suppose we are doing the email register module, then the hierarchy like below. Means identify all such possible exception that may occur while executing the application. Then create the exception class for that reason and if any one fail then don't throw the technology specific exception or persistence tier exception, wrap them into business specific exception and throw to the user. So that the user feels comfortable to use the application.

  ApplicationGenericException---application exception

  EmailRegisterException----module exception

  Email invalid Exception----component of module

  PasswordInvalid exception----component of module



- So when the delegate receive the exception from the dao it identify the exception and wrap the technology specific exception into one of the business specific exception and send back to the action class , so that action class convert this exception to an user friendly exception and send back to the user.

Question--- so interview will ask the question--- how did you solve the exception as part of your project----
Answer----actually we are believe in the prevention rather than curing the problem. 1st we are identifying what such possible exception that may arise in the application , so that we can avoid the problem before it arise. After identifying such problem then we are prepare for handling such exception.

- Lets talk about the previous use case , that the email registration process, here we are 1st identifying the problem like the email may insert properly or may not be insert properly , so that before forwarding the request to the delegate we are pre validating the email is correct or not .
- So that we can assure in some extend that the request can be processed easily , then if any issue further came from the dao then we are applying the business specific handling exception process that we are discussed earlier .So 1st we are try to prevent the error , but we can not avoid all the possibility of the failure, so we are identifying all such known exception(checked exception are known generally) and we are handling such exception by converting them into business specific exception.
- But while executing the runtime exception or the any unknown error may arise, for that we are configuring one global error page as part of the web.xml .so in all the way we are avoiding in showing ugly error page back to the user by throwing may business specific error page or by showing global error page so that the user feels comfortable. this is the reason why we are using business delegate .we are using business delegate for handling such exception.
- Question----where you are handle the exception, means at the client side or at the server side? Which side you prefer
- It depends on the nature of the application. If the application seems to be an intranet application , then it is

recommended go for client side validation. Bz as the application will be used by the organization itself the chance of misusing the application will be less.

- But if the application is publicly accessible means enterprise application then it is recommended for server side validation , we can implement at client side , if we are implementing client side then one intruder may cause problem by directly accessing the server instead of the jsp page, then the application may arrive at more no of exception, so there may a chance that the application may broken.

Question- how do you give name for delegate and dao in your application & how many delegate and dao you wrote in your application-----

- We are creating the action class per request or operation or per the functionality of the application, means for every functionality we are creating one action class. If we have 30 functionality then we write 30 action class.
- But we are not creating the delegate class based on the functionality of the application. We are creating one delegate per one module in our application. If 5 modules are there in the application then 5 delegate we are creating. If the module is big then we are dividing into multiple sub module and we are creating one delegate per one sub- module.
- How you name the method of delegate---we are not naming the method based on the persistence operation, we are giving the name based on the business functionality or the business operation that we are performing.
- We are not creating dao based on module or functionality, rather we are creating the dao class based on the group of relationship between the table. Means based on the relationship between the table we are creating dao.

Question-----how do you manage transactionality in your application-----

- People are telling we are managing the transaction as part of the dao level. But it is not correct so lets discuss what is the problem. Lets take there are 2 dao is there Company dao and job dao , when delegate call dao we should 1st store the company data , take the company id using auto generated using prepared statement and then store the job dao with the company id.
- Here 1st delegate call company dao , pass the company BO data to the dao , dao perform the operation by creating the connection. And after completion it commit bz we are managing transaction as part of the dao level. Then it forward the response to the delegate then delegate call job dao by passing the company id. Then job dao trying to perform by creating one more connection. While executing it fails in execution it goes to finally block and we are checking boolean is true or not, as it is fails it is still false so we are rollbacking the operation.
- But the company data is already committed , so it is not call transactionality we should completely commit all the

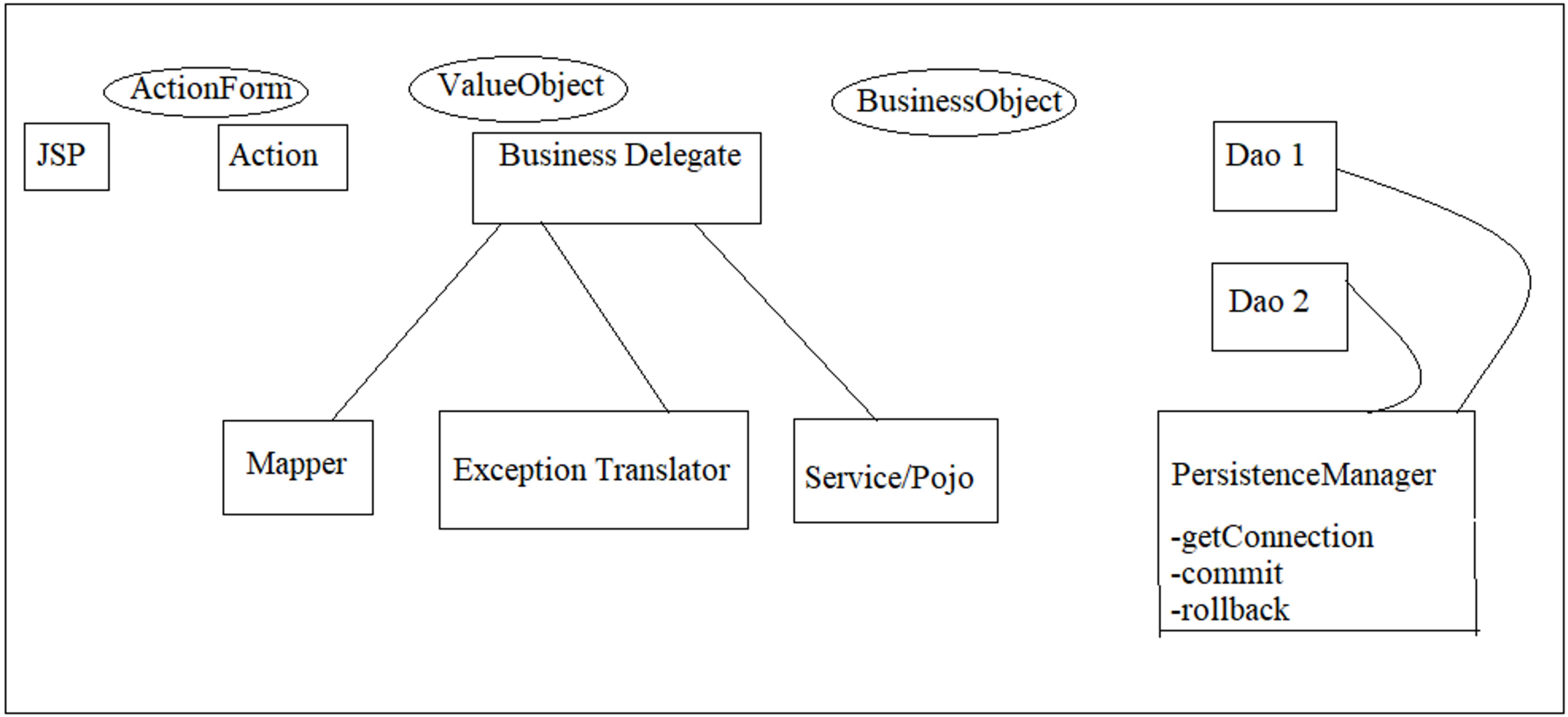
operation or completely rollback the operation, but it is not possible if we are managing transaction as part of the dao.so manage transaction as part of the delegate level.

- So 1st create the connection in the delegate and call the dao by passing company BO and connection as an input, so that it perform the operation and return the company id , then delegate call the job dao by passing the same connection , company id and job BO as an input . So that it perform the operation if it completely successfully then control comes to delegate here we are committing . So commit then both data will be persisted. Lets see the failure path
- We are passing creating connection to company dao it execute successfully then it return company id to delegate . Then it call the job dao by passing same connection, company id , job dao then while performing the operation it fail then it throw the exception back to delegate bz we are not managing transaction as part of the dao . So as exception raised at delegate we are checking in finally wheather the flag is true or not , it is still false bz the try block is fail in execution , so we are rollbacking on the connection, then both the operation is completely rollback as we are using same connection for both the operation. So it is the 2nd reason why we are using business delegate in our application. we are using delegate for managing transactionality.
- When ever action class receive the data in terms of action form it has to send the data to the delegate when it forward the request to the delegate , but it should not send the data in terms of action form , bz action form is the technology specific so if the delegate is receiving the data directly in terms of action form then it tightly coupled with the persistence tier technology.
- If we change from struts to spring then we have to modify the business delegate, so don't pass the data in terms of action form rather convert the data into ValueObject and pass it to the delegate. ValueObject will acts as a carrier of the data between the persistence tier and the business tier of the application, so that the business tier is decoupled from the persistence tier.
- The ValueObject is exactly contain same data type and no of attribute , that present in the action form . So the value object is the exact copy of the action form. But don't convert the value object to business specific data in action class , you just copy the action form to value object in the action class and all the data type should be string only.
- Let the delegate should responsible for converting the value object to the business specific data. Bz if you convert it in the action class then the persistence tier will be tightly coupled with the business tier, any change in the business tier will affect the persistence tier.
- so finally the data received by the delegate in terms of value object ,value object is a java bean so it should be implements from serializable.bz in cluster environment it should work. but we cant pass the value object to the dao directly . Bz the data type of all the attribute inside the value object is string only. The data type of attribute

corresponding column in the data base may be in other format like int or float .

- so delegate should be responsible for converting the value object to business specific data , lets take the previous example that we want to store company and job data. Then the value object contain both the company data and the job data, but company dao only want company data. So delegate should pass to dao only the company data, so here BusinessObject comes into picture.
- Let the Delegate convert the ValueObject into corresponding dao data in terms of BusinessObject and the attribute from string to corresponding data format , so it can pass to the dao. So it is the third reason why we are using business delegate.
- So if you are performing all these operation in business delegate then where do you write your business logic. So we have to tell that as the project is a moderate project so, we are writing the business logic as part of the delegate itself, as there are no such complex logic is there. But if your project is big and logic is much complex then we should explain the architecture as follows. here we are managing all this logic in delegate , but there it should be in different class.

Real world architecture



- Whenever we submit the request from the jsp page request comes to my action class , action class receive the data in terms of ActionForm .
- When ever action class receive the data in terms of action form it has to send the data to the delegate when it forward the request to the delegate , but it should not send the data in terms of action form , bz action form is the technology specific so if the delegate is receiving the data directly in terms of action form then it tightly coupled with the persistence tier technology.
- If we change from struts to spring then we have to modify the business delegate, so don't pass the data in terms of action form rather convert the data into ValueObject and pass it to the delegate. ValueObject will acts as a carrier of the data between the persistence tier and the business tier of the application, so that the business tier is decoupled

from the persistence tier.

- The ValueObject is exactly contain same data type and no of attribute , that present in the action form . So the value object is the exact copy of the action form. But don't convert the value object to business specific data in action class , you just copy the action form to value object in the action class and all the data type should be string only. Then pass the ValueObject to the delegate.
- to convert the ValueObject to the corresponding Business functionality specific object means to BusinessObject , it should call the mapper class . As the delegate is meant for delegating , it should not perform any business logic , earlier what we have discussed is for small level project so there it is correct. The mapper class should be responsible for converting the ValueObject to the BusinessObject and pass it to the Delegate.so that delegate can call the service class and perform the operation. That's how we can manage in converting data.
- How to manage Exception-----delegate will pass the BusinessObject data to the service class , whatever the business logic service class perform , if Service class want to perform some data base operation then he should not call directly the dao class , it should pass the request to the delegate , delegate should be responsible for calling the dao,

- While executing if dao fails in executing then it should throw the exception back to the delegate. but here in this architecture delegate should not manage the exception. Delegate should catch the exception whatever the exception will be thrown by the dao, and delegate should pass the exception to the exception translator class , translator class after receiving the exception it should convert the technology specific exception to the application specific exception. And pass the Application specific exception back to the delegate , so delegate should throw the exception back to the action class , and action convert t this user friendly error page. That's how we manage the exception as part of our application.

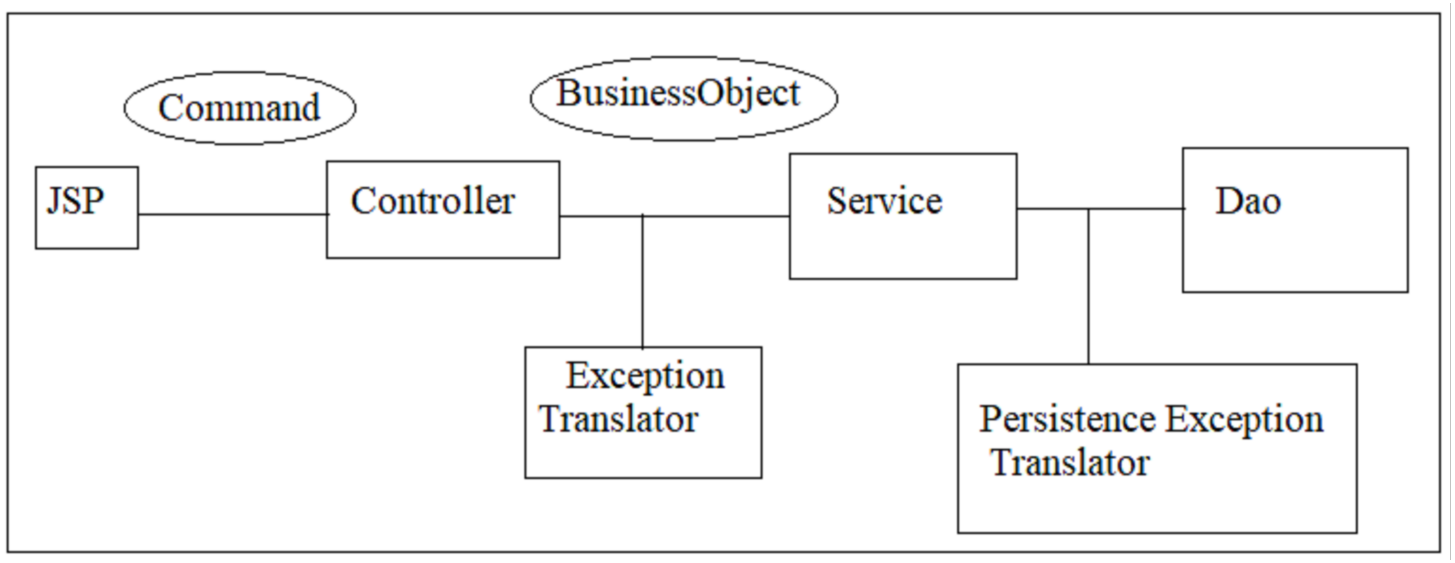
- How to manage transactionality -----previously we are managing transactionality in delegate nut actually it should not.so where to manage lets talk.
- If Service wants to perform the persistence operation, then it call delegate b passing Business Object then delegate call the method on Dao1 1st time ,previously we are creating the connection in the delegate but here it should not , after calling method on dao1 , dao1 should go to the Persistence Manager requesting for a connection, then persistence manager will create one connection for dao1 and before giving the connection to the dao1 , persistence manager should store th connection as part of the current ThreadLocal .
- then it gives the connection to dao1 after dao1 complete its execution , it pass the control to the delegate then delegate call the method on dao2 , then dao2 should go to persistence manager for getting the connection , then

manager will check in the thread local wheather connection is already created for this current thread of execution or not , as it is already created it gives the same connection to the dao2 , then dao2 perform the operation after it pas the control to delegate .

- Then delegate will check wheather any exception arise or not if arise then it 1st call the rollback method on the persistence manager on the same connection then completely operation will be rollbacked , then it pass the exception to translator. If there is no exception delegate call the commit method on the Persistence Manager and whole operation will be committed , that's how we are managing the transactionality as part of my project.

- If we are consuming any service from the web-service then the service class will not be call the provider, service will pass the request to delegate , then delegate will call the service locator for the stub or port object , then service locator should be responsible for creating stub object and pass it to the delegate and then delegate should call the provider. If you are want to expose your business logic to external world then expose your service or pojo class to the external world using contract last interface.

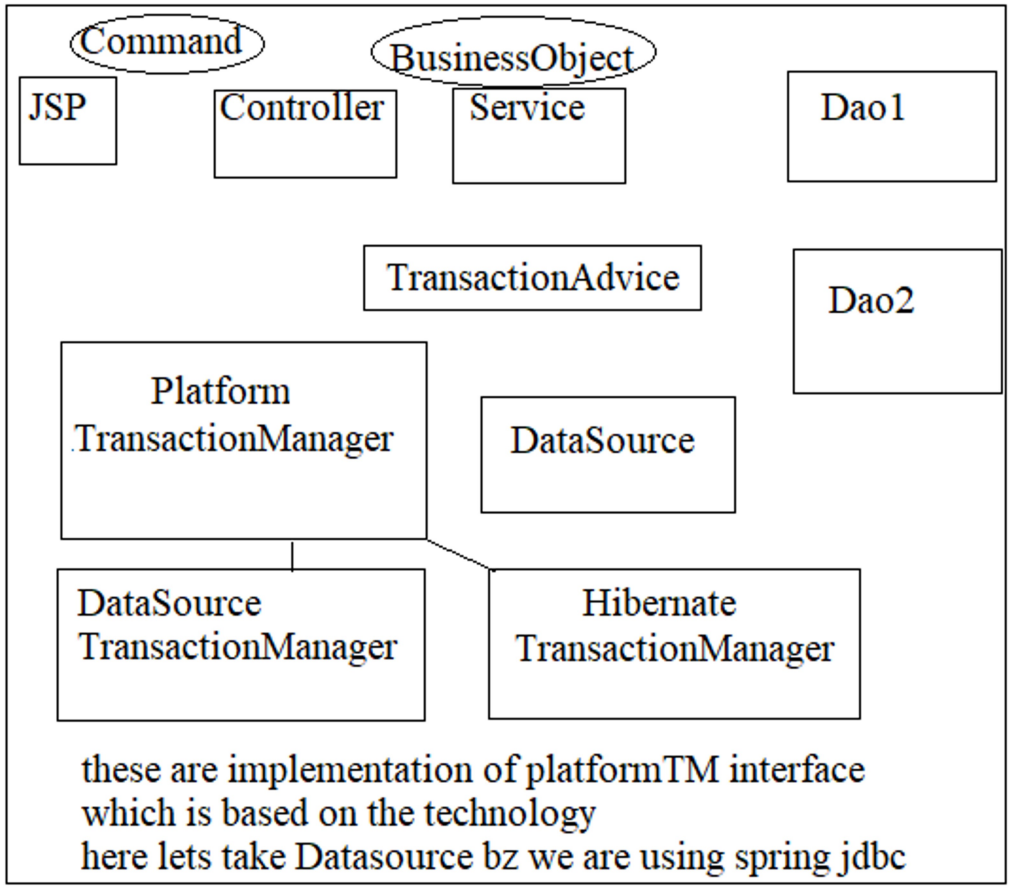
Spring Architecture



- Whenever we submit the request from the jsp page the request received by the controller , the controller reads the data in terms of command, then the control received by service class and service class call the dao and perform the operation . This is the overall architecture. Here delegate is not there so who perform the delegate work lets talk.
- Exception management------ whenever the service class calls the dao class to perform some persistence operation , if dao fails in execution how to manage exception lets understand-----in spring what ever the exception that raised in the persistence tier , means it could be database specific exception or it may be technology specific exception , whatever the exception is raised as part of the dao that exception is not thrown by spring framework to the service class . All the exception that raised all are converted to spring jdbc specific exception.
- Whenever the exception is raised in the dao before it throw to the service class , in between dao and service class there is one more spring component is there called persistence exception translator , which convert the technology or database specific exception to the corresponding spring jdbc exception.
- As all the spring jdbc exception are runtime exception or un checked exception so we don't need to catch the exception in the service class , then the exception will received by the controller , inside the controller convert the spring jdbc exception business specific exception and friendly error page.
- But in the service class ,if the business logic fail in execution , as we are using various api then this api are if fail in execution then this may throw the checked exception . These known exception where we have to map lets talk. As the delegate is not there.
- When service class throw the exception the exception receive by the controller, but if controller write the logic for mapping the exception then the presentation tier will be tightly coupled with the business tier specific api, so don't write it in controller so where to write.
- Here use aop , and advice on the service class method, whenever the service class fails in execution then the exception will be thrown to controller, as we are advising the throws advice on the service class before throwing the exception after throwing method of the throws advice will be called , here map the exception to the business specific exception and throws to the controller so that the controller will be able to convert it into business friendly error page. if any other unknown or runtime exception raise then we are mapping them as global error page in the web.xml.
- Data Conversion----when the request comes from the jsp page then the controller will receive the data, in terms of command, can we pass the command to the service class or not, yes we can pass the command to the service class bz command is a pozo class which is technology independent , but it is recomeneded to convert the command to business specific data.
- So here the Business Object comes into picture , copy the command class object to the BO and send the BO as an

input to the service class. Question is what is the data type contain in the attribute is it only string or any other , spring support any data type you want , bz while wrapping the jsp input data to the command if any type conversion error raised spring able to solve the error, it not throw ugly error page to the user like strauts throw. So in command we can put any premitive data type not only string.

- Transaction-----



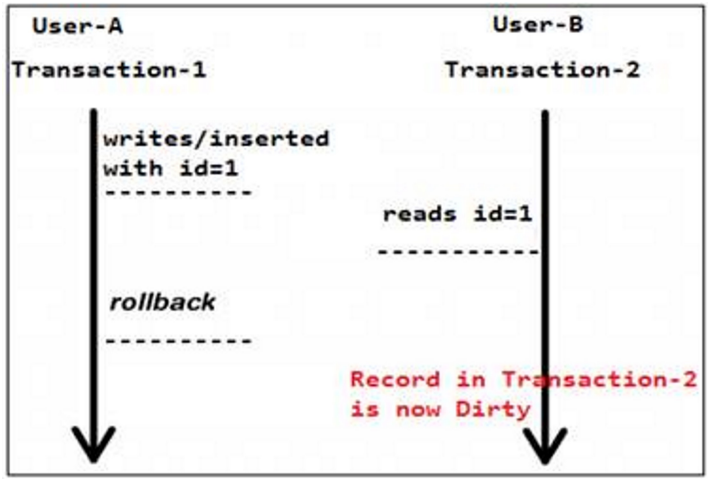
- Whenever controller delegate request to service class as we are working on transaction then the transaction manageability should be in the hand of the service class , so before calling the dao1 service class should call the DataSource Transaction . Begin Transaction and , then only call the dao , and whenever the dao successfully complete then it has to commit , but it is not in one service class we have to write , in all the service class of our application we have to write, so it is a boiler plate logic.
- So go for aop , and advice the service class with around advice so that whenever the service class method called it

should begin an after completion it should commit, and advice the throws advice if fail then it should rollback before throwing the exception.

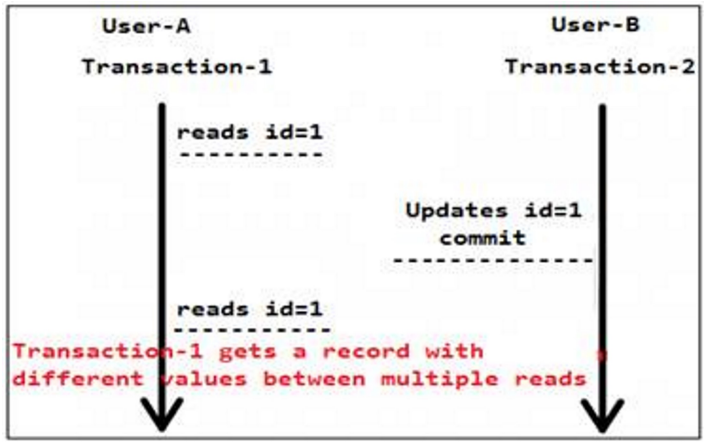
- As it is common for all the service class so spring provide one advice class called TransactionAdvice which has 2 method one is invoke which is around advice , and another is after throwing which Is nothing but the throws advice.so advise the service class with TransactionAdvice so that both will be applied.
- Whenever the control comes into the controller , it call the methods on the service class, as we are applying transaction advice so before calling service class method the control comes into the invoke() method of the transaction advice class , here we are beginning the transaction , as the data source is with the Data source transaction manager(in case of spring jdbc) it tell to the data source whatever connection you create for this transactional boundary , create the connection with auto commit as false,
- After begin the transaction here we call the proceed() , so the control goes to the service class method, the method called the dao1 , then the dao1 goes to jdbcTemplate for the connection, jdbcTemplete goes to the data source for the connection, then dataSource create the connection with auto commit as false go to the DatasourceTransactionmanager and store in the thread local of the transactionManager and give the connection to the jdbcTemplate, then dao1 perform the operation , after the control comes into the service class.
- Then the service class call the dao2 , dao2 go to jdbcTemplate , jdbcTemplate go to the data source for the connection, then data source go to the transaction manager check in the thread local wheather it already created or not for this current thread of execution, as it already created then it return the same connection to the dao2 , it perform the operation and after control comes to service class , then control comes to the proceed() statement of the advice class , then here we have to commit on the data source transaction manager , and then after return the data to the controller.
- Lets see the failure path, lets take transaction is already begin, the control is there with the service class , it call dao1 it take the connection from the jdbcTemplate and then service calls the dao2 and dao2 fail in performing so the exception is raised before throwing the exception the after throwing() of the transaction advice is called and here we are rollbacking the transaction in data source transaction and throwing the exception so like this we manage the transaction.
- We can specify on which of the method we have to apply the transaction by specifying attribute in configuration file.

Isolation Problems

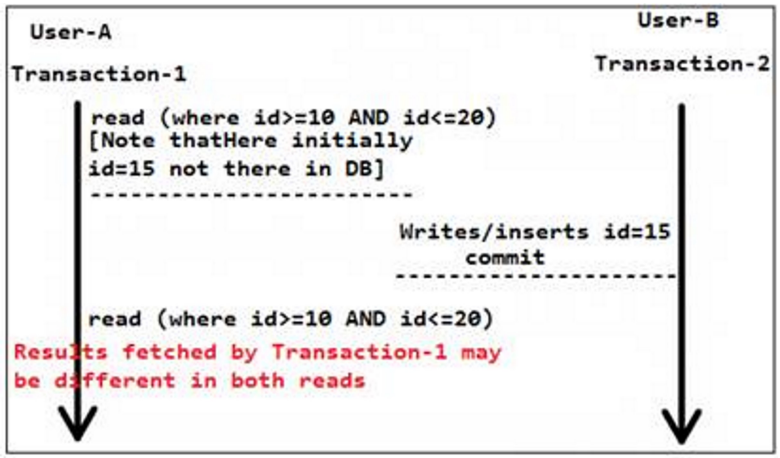
- Dirty-Reads Problem
- Non-Repeatable Reads Problem
- Phantom Reads Problem
- Dirty-Read means here user-A write some data in the table within transaction1 as the transaction1 is still active ,it is not committed. Within active transaction1 , transaction2 started by user-B , and he may read the data that is inserted by transaction1, but after some time transaction1 fails and it is rollback, so that whatever the transacton1 modified that is with the transaction2 so that are the dirty data. And this is called dirty-read problems.



- In this example Transaction-1 reads one record from the table. Then Transaction-2 writes data by updating that same record and commits on the same table. Later Transaction-1 reads that same record again and ma get different values because Transaction-2 made changes to that record and committed. This is a non-repeatable read. This is occurring bcz of we don't have any row level locking in an concurrent access/updates.



- If a transaction reads one record from the database multiple times the result of all those reading operations must always be the same. This eliminates both the dirty read and the non-repeatable read issues, but even this way other issues may occur. Let's see .



- In this example Transaction-1 reads a group of records. Meanwhile Transaction-2 inserts a new record in the same group that Transaction-1 initially fetched and commits. Later Transaction-1 reads the same range again and will also get the record that Transaction-2 just inserted. This is a phantom read: a transaction fetched a range of records multiple times from the database and obtained different result sets (containing phantom records).
- Phantom means hunting new one or finding new one. This is occurring bcz of we don't have any table level locking

in an concurrent access/updates/inserts.

Isolation Levels

So we can solve the problem by this applying these levels.

1. READ_UNCOMMITTED
2. READ_COMMITTED
3. REPEATABLE_READ
4. SERIALIZABLE

READ_UNCOMMITTED:

- ✓ If we use this isolation level then we have all the 3-inconsistency problems will occurs. Bcz Transaction Manager is going to start the transaction in an READ_UNCOMMITTED isolation mode. That means it will reads all uncommitted data also.

READ_COMMITTED:

- ✓ Used to avoid Dirty-Reads that means Transaction manager will begins transaction in READ_COMMITTED isolation level. That means Transaction Manager will asks the DB to read the data which is only committed data not uncommitted data.
- ✓ It avoids Dirty-Reads problem but still we have Non-Repeatable Reads Problem and Phantom Reads Problem bcz we are not locking any record level locking or table level locking.

3. REPEATABLE_READ:

- ✓ It avoids Dirty-Reads problem and Non-Repeatable Reads Problem bcz it acquires the row level lock or record level lock but still we have Phantom Reads Problem.

4. SERIALIZABLE:

It is the high level isolation which will acquires table level lock.

- ✓ SERIALIZABLE isolation level is the most restrictive of all isolation levels. Transactions are executed with locking at all levels (read, range and write locking) so they appear as if they were executed in a serialized way. This leads to a scenario where none of the issues mentioned above may occur, but in the other way we don't allow transaction concurrency and consequently introduce a performance penalty.

	Dirty-Reads	Non-Repeatable	Phantom-Reads
Read-Uncommitted	Yes	Yes	Yes
Read-Commited	No	Yes	Yes
Repeatable-Read	No	No	Yes
Serializable	No	No	No

How do u manage the transactions across the Dao's?

Ans: Using Service Layer.

How do u manage the transactions across the services?

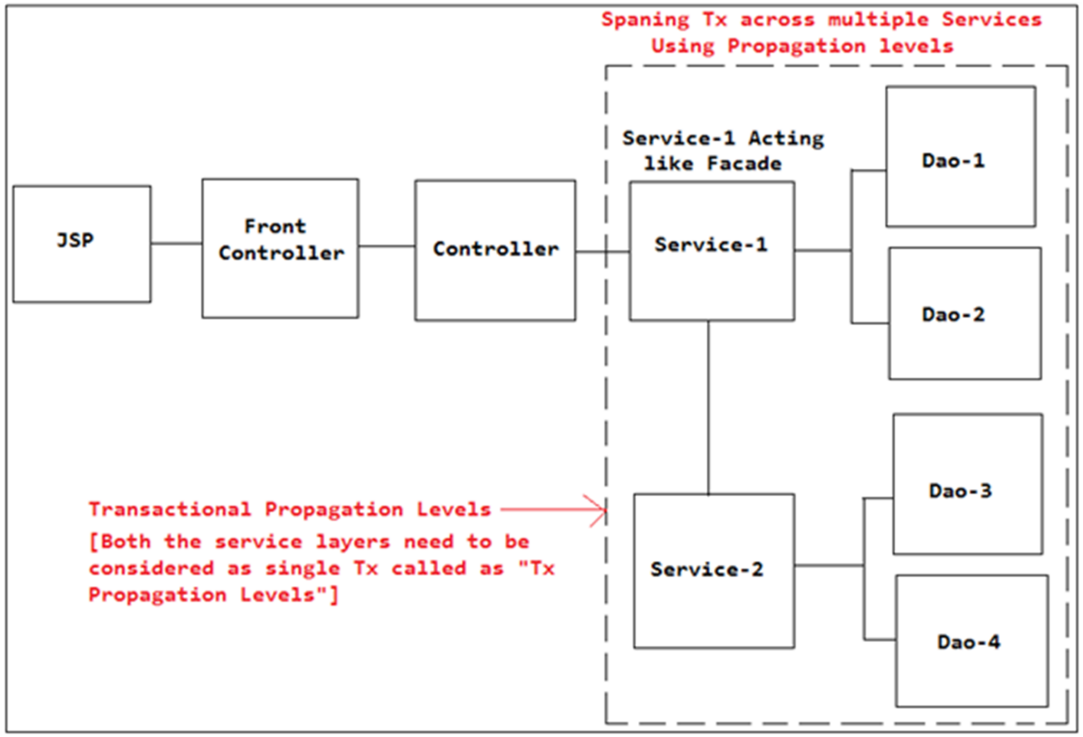
Ans: Using Transactional Propagation Levels.

- ✓ REQUIRED – Support a current/same transaction; creates one new active transaction if there is no active transaction in the current thread of execution.
- ✓ REQUIRES_NEW – Create a new transaction, suspending the current transaction if one exists. That means it always requires a new Transaction for each service.
- ✓ SUPPORTS – if already there is a active transaction then it will participate in the current transaction, if not execute non-transactionally , it will not create one new transaction..
- ✓ NOT_SUPPORTED – Do not support a current transaction; rather always execute non-transactionally.

- ✓ MANDATORY – Support a current transaction; throw an exception if there is no current transaction exists.
- ✓ NEVER – Do not support a current transaction; throw an exception if a current transaction exists.
- ✓ NESTED – Execute within a nested transaction if a current transaction exists, behave like REQUIRED else. After the 1st transaction if we want to start one new transaction then if the 2nd transaction is failed , then until the last save point will be roll backed means before the save point will not be roll backed.

Flow:

So when the controller is calling the register() method on the service-1 layer then method call will not enters into the register() instead of the control goes to the Tx Aspect tx:advice it checks what is the read-only is false or true then Tx isolation and Tx Propagation level and begin the Tx according to that.



Understanding with Combinations:

The combination given below are explained based on the **Diagram: 3**

Case: 1

Service-1 method: REQUIRED

Service-2 method: REQUIRED

- In this case if service-1 has active tx then service-2 also join within the same tx so if any exception raised within the service-2 Dao classes then all will be rollbacked. As they are participate in same transaction.

Case: 2

Service-1 method: REQUIRED

Service-2 method: REQUIRES_NEW

- In this case if service-1 has active tx then service-2 will suspends (suspends means it will continue with new tx in the service-2 and service-1 tx will applicable in service-1 only) the current tx and begins one more new active tx to perform the operation. So if any exception raised within the service-2 Dao classes then Service-2 only will be rollbacked but Service-1 will gets committed. Loan Approval is normal actual Tx and Audit will be the another Tx where we need to use this Tx Propagation level. That means if Audit throws (Service-2) exception fails also the actual business operation (Service-1) should be committed.

Case: 3

Service-1 method: REQUIRED

Service-2 method: SUPPORTED

- In this case if service-1 has active tx then service-2 will neither joins within the current tx nor begins new tx rather executes non-transactionally without any tx by suspending the current tx of the service-1.

Case: 4

Service-1 method: REQUIRED

Service-2 method: NEVER

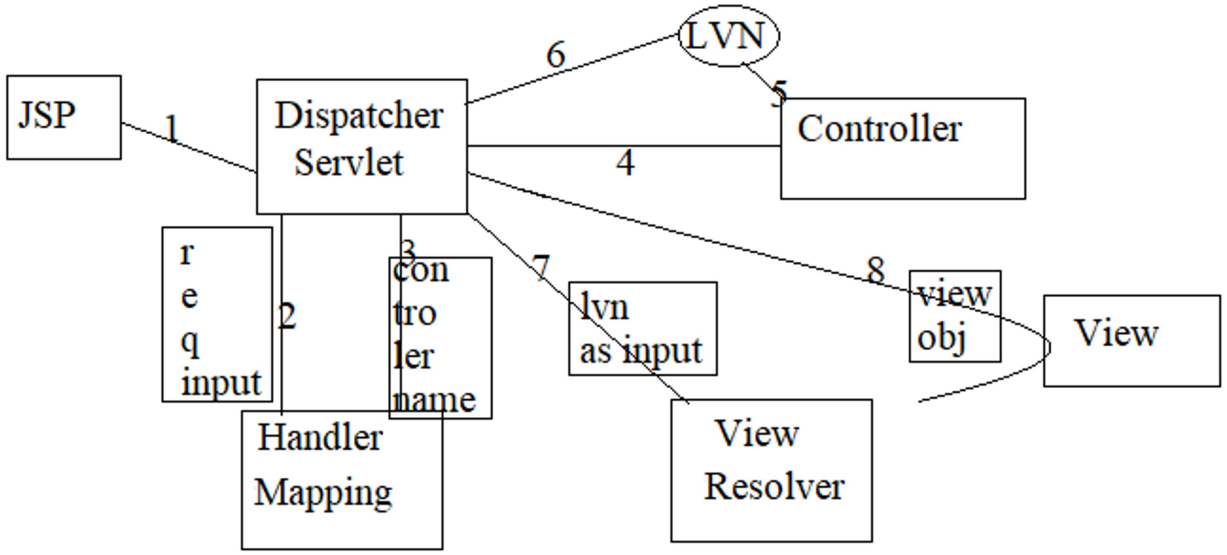
- In this case if service-1 has active tx then service-2 will neither joins within the current tx nor begins new tx and will not executes rather it will throws exception stating that call me without any tx.

- In most of the cases, we may just need to use the REQUIRED. If we didn't configure any propagation level by default is REQUIRED.

Question---------What happens by default when an exception raised at the application , it roll back or not?

- It depends on the Exception, if the Exception that raised is runtime or un checked exception then it roll back, but if the exception is Checked Exception as their sometime alternate path of execution is there it will not roll back by default , so that some data may committed some may not. We can specify for which checked exception the transaction will be rollbacked,

Spring MVC flow



A. Whenever the user send the request from the jsp page, the request will received by the DispatcherServlet , here it acts as a front controller , upon receiving the request it apply the common system services for all the request that are coming into my application. And apply the additional logic and the validation logic on the request, and it perform th Command management and View Management . Then it has to delegate the request, as it can not perform the

business logic bz it an view component .

B. So controller comes into picture, but it don't know who is the corresponding controller to whom it has to delegate the request , bz in the application for every functionality we are writing one controller, so handler mapping comes into picture.

C. DispatcherServlet pass the request as an input to the HandlerMapping , so HandlerMapping identify the corresponding controller information and give the information or name of the controller back to the Dispatcher servlet.

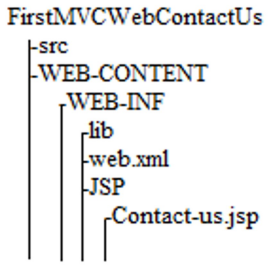
D. So DispatcherServlet delegate the request to the controller, then controller perform the request specific business logic , and after completion it has to send the response back to the user, but if controller is send the response back to user , it will tightly coupled with the view component, as spring support multiple technology as view component. If view technology change the controller has to be modified.

E. So controller send the logical view name back to the DispatcherServlet , it is the responsibility of dispatcher to identify and render the corresponding view for that LVN. But the dispatcher don't know who is the view for that LVN.bz the view is something that is written by the programmer. So here viewResolver comes into picture.

F. DispatcherServlet pass the LVN as an input to the ViewResolver , resolver identify the view and instantiate the corresponding implementation view object and give the view object back to the DispatcherServlet . And dispatcher call the view and render the response page back to the end user. As spring supports multiple view technology , so to decouple from the view technology it provides one interface called View Interface.

G. All the view classes are implemented form the View interface so that we can switch from one View technology to another view technology , without modifying the application.

❖ When it comes to spring mvc based project , it is recommended to put the view classes inside the WEB-INF , bz it is the protected directory of the application , any one can not access directly the view classes from the WEB-INF. Only the other component of the application like Servlet , controller can access this classes. Don't put the views directly inside the web- inf put those in a folder, this is the convention. There are 4 reason why we don't put the view inside the WEB-CONTENT.

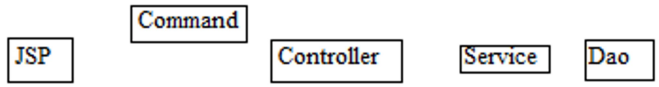


- 1st-reason------ To hide the corresponding view technology from the end user means--suppose we are using jsp as view technology , so if it is in web-content then the user can directly access the jsp pages . After sometimes we change from jsp to velocity or some other view technology, then the user is friendly with jsp , so he has to access the velocity, so the user experience got impacted. So it is recommended to put the view pages inside the web -INF so that the user can not call the view, it will call servlet or controller and controller call the views.
- 2nd-reason------There are some pages that has to render based on the outcome after the post processing of the request , means that pages need data as an input which can be passed to that pages after performing some operation, but if it is under web-content then user directly can access the pages , as the pages need the data ,and the data is not there then the pages run into error and show the ugly error page to the user. So don't let the user directly access the view pages.
- 3rd-reason------hide the view technology to end user- means if the user know the view technology, suppose we are using the jsp as view technology , when the user is trying to access the application then when the response page is comes from the server, then at that time one intruder can write some java script logic as part of the jsp page in the response , so that he can hack the user data by writing some event logic at the server side of the intruder. So don't let the user to know the view technology.
- 4th-reason------Hide in direct accessing the jsp page, so that the user information is safe, suppose user is trying to book a ticket in the book my show , after booking the ticket he download the ticket and without logout he move, but the data will be there in the user session, so some other person can check the history and directly access the jsp page, but if the jsp can not be access directly then the user has to go through the servlet but the data will not be there in the user session.
- How to build the application------ as all the component of spring mvc are has to be called by the Dispatcher servlet ,so let the dispatcher servlet itself will create the object of the component, but the problem is for every request it has to create the object for calling. So let the ioc container will create the object for those classes. And let

the ioc container has to be created by the dispatcher servlet , so that every time it will go to the ioc container and will use the component,

- When first time the dispatcher servlet is created it call the init method , inside the init method it search for the file called dispatcher-servlet.xml which is the spring bean configuration file, so name your spring configuration file as servlet name-servlet.xml and place it inside init parameter of the web.xml. So it will be created by the dispatcher servlet.
- to call the controller the dispatchers servlet has to go to the handler mapping so we need handler mapping, so spring see in this directory K:\BOOK\SRIMAN SIR\Framework notes\IMP_NOTES\Spring\7 MVC\Material\2 HandlerMapping , provide all the component information in the Spring bean configuration file. And let the dispatcher servlet creates the ioc container , so at the time of creating the dispatcher servlet , the init method called and the ioc container will be created, then all the spring provided functionality will be applied.

- So lets talk about the spring mvc architecture



- Here jsp forward the request to the controller and controller to the service and service to the dao. We are providing the service and dao as bean in the ioc container, bz to talk with dao service needs dao object , so dao is inject ono the service, and to call service controller needs service object so inject service into the command, as the ioc container is created by dispatcher servlet inside the init method, so if we dot want to use spring mvc as presentation tier, then the along with controller the dispatcher servlet will also gone, bz those are spring specific component.
- As the ioc container is created by the dispatcher , so if dispatcher is gone then the service and the dao will also be gone, so my business tier also gone. So don't create the ioc container by the dispatcher servlet. Lets create the dispatcher servlet create the servlet name-servlet.xml file, but the application-context .xml will not be created by the dispatcher.
- So if dispatcher will not create application-context.xml ioc container so who will create the ioc container, here

ContextLoaderListener comes into picture. As the controller need service to be injected into him, so 1st the service and the dao should be created, so 1st create the application-context ioc container then create the dispatcher-servlet ioc container (I spring mvc there are 2 ioc container) .

- But the problem is dispatcher -servlet ioc is creating at the start of the deployment of the application, so we have to create the application ioc container before any of the servlet component or dispatcher ioc gets created. So at the time of deployment 1st the servletContext will be created , then create the application-context ioc container by and add in the ServletContext so that it will be created before the Dispatcher-servlet ioc container gets created. So that all the business tier specific component will be used by the dispatcher ioc container, if spring mvc is changed by some other presentation technology my application-context ioc container will be still there.

Internal Request Flow

- When we are configuring dispatcher servlet we can configure ContextLoaderListener in the web.xml also, it is not mandatory , if we want business tier in another ioc container then create context loader listener . So what happen if we configure context loader listener.
- At the time of deployment of the application , the application server will start deploying the application by reading the web.xml , so 1st the war packaging structure get validated, after it goes to the web.xml and validate the content of the web.xml against the xsd. After it read web.xml configuration completely and create ServletContext object with all the meta data of the configuration.
- ServletContext is not a object to store the data , it is a global object that is getting created for holding the information of one whole application. So it is the only single point from where we can get all the information about the application. So when the application server will able to read the web.xml and create the ServletContext object then only we can say the application deployed.
- Then after servlet container goes to the meta data of the ServletContext and looking for classes are object of type listener , so 1st listener , 2nd filter ,3rd servlet will created in this order. As we are configuring the ContextLoaderListener and it is of type ServletContextListener type, so servlet container will create the object of the ContextLoaderListener and it will call contextCreated(ServletContext) method by passing ServletContext as parameter, and inside this method it get the ContextConfigLocation from the ServletContext . Means it get application-context.xml from the context param value , and it create the ioc container of one of the type of

webApplicationContext .

- Context loader listener will create one ioc container, and it is of type web application context(it is an another interface extending from the application context, and it has several implementation XmlWebApplicationContext , AnnotationConfigWebApplicationContext) .why we will talk later.
- After context loader listener complete operation then control comes to servlet container , then it looks for filter then for servlet, we configure servlet, so it check we configure load on startup or not, as we wrote load on startup as 2 so it understand that before this some other has to be created. So it 1st create listener then it start creating dispatcher servlet.
- After creating it call the init method on the dispatcher servlet, inside the init method it read the servlet name from the ServletConfig and append "/WEB-INF/" before and -servlet.xml after servlet name and create the name of the spring bean configuration file, and with this file it create one more ioc container called web application context implementation object .
- so in mvc both context loader and dispatcher will create ioc container of type WebApplictionContext type. As those are dependent each other bz we have to inject business tier component into the presentation tier which are part of 2 ioc container, so we have to nested the ioc container.
- So while creating the dispatcher ioc container pass the application ioc container as parent ioc container(Context loader listener ioc) , so the child ioc (dispatcher ioc) can see the parent ioc container. But the parent is created by listener so it is with listener so how we can pass the parent into child.
- So after context loader listener create the ioc container in the contextCreated method , it place the ioc container in the ServletContext object . With name as WebApplicationContext.class as key, and ioc container object as value.bz the only way to share the web application component in j2ee is through the ServletContext. . So after context listener completed operation , servlet container instantiate the dispatcher servlet. And called init method.
- So inside the init method it will call getServletContext() and it get the ServletContext , and from the ServletContext it will get the parent ioc container of type WebApplicationContext , by calling context.getAttribute(WebApplicationContext.class), and after while creating the child ioc container means the dispatcher ioc it pass the parent ioc container and create the child ioc container, that's how the ioc container are nested.

```
class ContextLoaderListener implements ServletContextListener{

        public void contextCreated(ServletContext context){
                String contextConfigLocation=null;
                contextConfigLocation=context.getParameter();

        }

        public void contextDestroyed(ServletContext context){}
}
```

```
class DispatcherServlet extends HttpServlet{

        public void init(ServletConfig config){
                String servletName=null;
                servletname=config.getServletName();
                servletName="/WEB-INF/"+servletName+"-servlet.xml";
               ServletContext context=null;
               WebApplicationContext  parentContext= null;

               context=getServletContext();
               parentContext= context.getAttribute(WebApplicationContext.class);

         }
        public void service(req, resp){
        }
}
```

- When we send request dispatcher servlet receive the request, it will go to ioc container and search for the handler mapping, as multiple beans are there , it is difficult to identify handler mapping bean. So it go to ioc and called context.getBean(HandlerMapping.class) , it will not pass the bean id for getting the object of handler mapping bz it don't know the id. We can pass not only bean id but also we can pass the class type as input so we will get array object.

  A a=Context.getBean("a"); here it return one object

  Object[] =Context.getBean(class type (A.class)); here it return multiple object so return type is array

- As we configure one handler mapping SimpleUrlHandlerMapping so it will get that handler mapping , of interface type as HandlerMapping , so we don't need to configure the bean id for the handler mapping bz it search with the

class type not with the bean id. Interviewer will tell how without configuring the bean id for the handler mapping dispatcher servlet get the object of mapping. So it called getHandler() by passing HttpRequest on the handler mapping, then it will return bean id of the controller, so dispatcher goes back to ioc container and identify the controller for that bean by Controller controller=context.getBean("bean id",Controller.class). Here also it design to interface , it will not bother which controller you wrote. And on this it call controlllr .handleRequest(req, resp);

- So the question is we wrote the class extending from the AbstractController class(recommended), and it contain handleRequestInternal(req,resp) method. But dispatcher called handleRequest(req, resp) , but this method is not there with our controller class so it goes to super class Abstract Controller class, as Abstract controller implements from the Controller interface and it override the handleRequest() .inside this method it is called simply the handleRequestInternal() .
- inside the handleRequest method it performs some request validation(if the request is get or post then it receive, if it put or delete it through the exception) , session validation. And call the requestHandleInternal method and it returns the logical view name to dispatcher servlet, the it go to view resolver get the view and ask him to render, this is the internal flow.

```
clas ViewAboutUsController extends AbstractController{

        public ModelAndView handleRequestInternal(req, resp){

        }
}
```

```
abstract Class AbstractController implements Controller{

        public ModelAndView handleRequest(req, resp){
                return handleRequestInternal(req, resp);
        }

        abstract public ModelAndView handleRequestInternal(req, resp);
}
```

Handler Mapping

- Handler mapping is the component which helps in mapping the incoming request to the corresponding controller. when we go for struts there are only one of mapping the request to the controller through the url. But in spring mvc it defines multiple ways in mapping the request to the controller.

SimpleUrlHandlerMapping-------this handler mapping helps in matching the incoming request to the controller , here it provide one properties in which we configure the url pattern with the bean id.

Advantage----as we don't configure the controller with the url pattern we can map one controller with multiple url pattern. We have the centralized configuration in one place , so that we can know for which request which controller ha to be call. We can easily trace the application flow.

Disadvantage------we have to write lot of configuration information.

BeanNameUrlHandlerMapping ------ here dispatcher servlet try to match the request to the controller , by matching the url pattern with the bean name of the controller that we are defining in the ioc container, so it is the responsibility of ours to write the bean name instead of id matching with the url pattern. This is the default handler mapping.

- When dispatcher servlet create the WebApplicationContext ioc container then , ioc container search wheather you define any spring mvc component or not, if not then it try to create the default component, BeanNameUrlHandlerMapping , InternalResourceViewResolver are the default spring mvc component.

ControllerClassNameHandlerMapping------here dispatcher servlet will try to match the incoming request with the class name of the controller. When it match it remove / and the .mvc or some other from the url pattern convert it to the class name format and match with the controller class name,

Disadvantage------if 2 controller with same class name of different packages are there then conflict will arise. An the class name are the technical name , if we provide this as url then the user feel discomfort in writing the technical name rather using business specific url.

ControllerBeanNameHandlerMapping-------here dispatcher servlet try to match the incoming url pattern with the bean id of the controller , so that the above 2 problem will solve. As bean id is unique across the ioc container, and we can specify the business specific id as url pattern.

Question ---can we configure multiple handler mapping or not in the ioc container or not?

Answer ----yes we can configure multiple handler mapping , this is called handler mapping chain, whenever the request received by the dispatcher servlet it try to map the request to the controller , to map it goes to one after another handler mapping and try to match , once it match the controller then it does not go to the next handler mapping. So don't configure one controller in multiple handler mapping , bz it always map in the 1st handler mapping . It don't go to the next handler mapping. So the question is why to configure multiple handler mapping?

Handler Interceptor

- ✓ If we want to apply some pre and processing logic for group of request then we need to go for handler interceptor. This helps us in applying the pre and post processing logic. But similar to interceptor we also have the filter in servlet and jsp. We can go for filter , why spring provide one more component called interceptor.
- ✓ Suppose we go for filter, then when we send the request the servlet container will receive the request, and call the filter(the scope of the filter is to the servlet container level) for applying the pre processing logic , and it apply the logic then the request receive by the dispatcher servlet , and it goes to handler mapping and try to find the controller for that request , but there is no controller associate with that request, so it is a bad request , for that also we are applying the pre processing logic. So this is the problem , if we go for filter , as the scope of filter is container level , for every request the pre logic will be applied, so it is waste for applying pre logic for bad request. So go for handler interceptor.
- ✓ whenever the dispatcher servlet receive the request it goes to handler mapping , match the controller for that request , then it try to check wheather you applied any interceptor for that request or not if not then it directly call the controller . So for configuring the interceptor we have to configure the interceptor with the url pattern, as we are already configure the url pattern in handler mapping , we can attach the interceptor with the handler mapping. We can configure separately but not recommended.
- ✓ So when dispatcher go to handler mapping for the controller, it identify the controller and check wheather you wrote any interceptor for that or not, if yes then it get the list of interceptor(we can apply multiple interceptor for one request) and the controller name and create one class object called handler Execution chain. And pass it to the dispatcher servlet.
- ✓ Dispatcher go to the class and get the controller name, before calling the controller it get the interceptor and call the pre handle method of the interceptor class, it return boolean value true or false if it return the true then it apply the

interceptor one after another by iterating in the list, if false then it don't apply the next interceptor . After pre handle it call the controller , after controller execution complete , it call the post handle method by passing the req, resp, LVN as input, if the post handle method complete its execution successfully then it call the after execution method , if post handle method failed then also it called after execution method.

- ✓ So we have to write the handler interceptor class by implementing from HandlerInterceptor interface and override 3 method preHandle() , postHandle() ,afterExecution(). That's why we write multiple handler mapping, for the group of the controller for which you want to apply the pre and post logic group them and write in one handler mapping and rest in another handler mapping.

Controller

When it comes to strauts there are only one class which acts as ac controller is Action class, which doesn't fit for all situation, but when it comes to spring it provide multiple controller for different situation.

- ❖ Whenever dispatcher servlet receive the request . DispatcherServlet don't knows how to handle the req bcz for every req different operation can be performed hence in order to handle the req we need Controller. We can write our own controller by implementing from Controller interface, but it is not necessary bz spring provides multiple controller for different different use case.
- ❖ ParameterizableViewController & UrlFilenameViewController

source	target
user trying to access directlry by clicking on hyperlink or entering url ----so no form submission require so i dont need ----RequestMapping, bz data is not send as part of the request, ----Data validation is also not require	showing static page

In this case we have to go for parameter and url file controller. Suppose user is access our web application , then we want show him our home page, then we don't have any requirement that the user submit the request by submitting form , and we don't have any dynamic page to show to user. Then we can go for this.

- ❖ Abstract Controller

source	target
user trying to access directlry by clicking on hyperlink or entering url ----so no form submission require so i dont need ----RequestMapping, bz data is not send as part of the request, ----Data validation is also not require	showing dynamic page

In this case we have to go for Abstract Controller. Suppose in the swiggy app the user added some food in the cart, and he close the application, after sometime he access our site, and he try to see his cart item, here he don't need to submit any form or we don't need any data from the user, but we have to show the cart by querying from the data base that user added previously, in this case we are showing dynamic data but user not submitting any data. So we can go for this.

- AbstractCommandController

source	target
user is trying to access by submitting data in the form ------data is send as part of the request -------i need Request Wrapping -------but i dont need Data Validation	showing static or dynamic page

Here we need to go for Abstract Command Controller. Suppose in flip cart app user is trying to watch some product, he sorted the order by category and brand and size of the product, here we need the data to show the product but we don't need any data validation, bz we are not performing any operation based on the user input, we are trying to show some product based on user requirement, if product are there then we are showing , if not there we are showing there are no product available for this search.

- SimpleFormController

source	target
user is trying to access by submitting data in the form ------data is send as part of the request -------i need Request Wrapping --------i need Data Validation	showing static or dynamic page

Suppose user is submitting data in the application for login, then here we have to validate the data wheather the user is valid or not, so here we can user SimpleFormController.

- MultiActionController

  Suppose user is trying to perform various operation on one functionality then we can go for this controller. Means adding a product , delete a product, or going previous page of product details etc, then rather writing multiple controller , we can use this controller.

  Question ----How to configure dispatcher servlet in spring mvc without web.xml

  Answer----

- Whenever we send the request the servlet-container receive the request, after receiving the request it try to find the corresponding servlet to whom it has to delegate the request , so we need to configure the information about our servlet to the container.
- By default servlet-container at the deployment time of the application , it goes to the web.xml, reads all the metadata and load in servlet-context , and then it scan the classes which are annotated with servlet specification annotation. And load into the servlet-context and create the servlet context object , so when it receive the request it goes to the servlet-context and find the corresponding servlet. So we have to add dispatcher servlet in the servlet-context as we don't have web.xml.
- We can bind the annotation, but if have the source code then we can bind, here we don't have the source code of the dispatcher servlet. Another way is write one class implementing from ServletContainerListener and there add the dispatcher servlet into the servlet context object. Bz after creating the servlet context object servlet container will called this Listener class by passing Servlet-Context object. So we can add.
- If we are adding the servlet through this process then there is no use , bz after servlet context object creation only it call this listener class, so that all the deployment life cycle process of the servlet container will be already completed. Suppose we are adding load on startup or we are configuring init param, all are not going to call by servlet container, as the servlet context object is created and all the initialization has been done. So there is no use of adding the dispatcher servlet. We have to add the dispatcher servlet during the deployment of the application.
- So in latest servlet specification they provide one interface called ServletContainerInitializer , so you write your class implementing from this interface and override the onStartup() method, inside the method add the servlet that you want to add, so at the deployment of the application servlet container read all the metadata and create the servlet context object before initializing any web application component it call this class and we can register our servlet

with servlet-container.

- So instead of implementing from this interface we can implement from spring provide classes called WebApplicationInitializer and this class internally implementing from one spring class DefaultWebApplicationInitializer and this class is implementing from ServletContainerInitializer . So at the time of deployment servlet-container will call the DefaultWebApplicationInitializer class bz this class is implementing from ServletContainerInitializer, and this class internally called our class which is implementing from WebApplicationInitializer. So that we can add the dispatcher servlet inside this class.
- Before when we are configuring the dispatcher servlet in web.xml, it is being created by servlet container , so we don't have the object of the dispatcher servlet so we cant add the ioc container to the dispatcher servlet , but in programmatic approach we can create the ioc container by our self and we can add into the dispatcher servlet. (here we are creating dispatcher servlet by calling new operator, people think we cant create but we can create, but if we create it will not call by servlet-container ,a s this is not register with the container).

```
public class DispatcherWebApplicationInitializer implements WebApplicationInitializer{

        @Override
        public void onStartup(ServletContext servletContext) throws ServletException {

                XmlWebApplicationContext servletApplicationContext=null;
                DispatcherServlet dispatcherServlet=null;

                servletApplicationContext=new XmlWebApplicationContext();
                servletApplicationContext.setConfigLocation("/WEB-INF/dispatcher-servlet.xml");

                dispatcherServlet=new DispatcherServlet(servletApplicationContext);
                Dynamic dynamicServletRegistration=servletContext.addServlet("dispatcher",dispatcherServlet);
                dynamicServletRegistration.setLoadOnStartup(1);
                dynamicServletRegistration.addMapping("*.htm");

        }
}
```

here we are creating the ioc container and we can add this ioc with dispatcher servlet

we can create servlet by calling new operator

here we are adding the additional component like load-on-startup or url pattern or init param

```
thid dynamic interface gives us     this
<servlet>
        <servlet-name/>
        <servlet-class/>
</servlet>
```

- If we are adding ioc manually then we don't depend on the dispatcher servlet default behavior.

AbstractCommandController

- Whenever the dispatcher servlet receive the request from the form page , it has to call the controller to call the controller it has to pass the command object, as the data is coming as part of the request. But the request wrapping into command object is not done by the dispatcher servlet. Bz the request wrapping is not a common logic for all the request so why dispatcher should do this wrapping.
- So dispatcher servlet simply goes to handler mapping and take the controller and call the handle request method . Bz it do not know which controller you wrote, it only know controller interface .

```
class AbstractController implements Controller{
        public ModelAndView handleRequest(req,resp){
                return handleRequestInternal(req,resp);
        }
}


class AbstractCommandController extends AbstractController{
        private String commandClass;

        public ModelAndView handleRequestInternal(req,resp){
                command=Class.forName(commandClass).newInstance();
                return handle(req,resp,command,BindException);
        }

        protected ModelAndView  handle(req,resp,command,BindException);
}


class MyCntroller extends AbstractCommandController{
        public ModelAndView handle(req,resp,command,BindException){}
}
```

- So the control goes to the top of hierarchy as the method is not there with our controller class, and there we are calling internal method , in the Abstract command controller we are wrapping the into command by reading request parameter, and this class calling handle method by passing the command object so dispatcher servlet is not doing the request wrapping.

Simple Form Controller

- Whenever we work on simple form controller , we have to use the spring form tag in our jsp page. So that in case of

validation failure we can show to the user back the same jsp page with the command object data, rather showing empty input control. if we are using simple html tag in jsp then , whenever the user submit the request from the jsp by submitting the form then the , data validation is happen. If the data is invalid then the dispatcher servlet take the command object data and put in the request scope and display the same page . But the page will have the empty input control it doesn't have data , bz html tag don't know how to read the data from the request scope.

- But if we are using spring form tag, those are not html tag , internally spring provide one one class for each tag, so that whenever the request data is invalid , the dispatcher servlet puts the command object data in the request scope, and as we are writing spring form tag, whenever the jsp page will convert to servlet and inside this servlet the _jspService method call , inside this the classes are executed for the corresponding form tag. Those classes are read the data from the request scope and convert the spring tag to html tag by reading the command object attribute value and put it in the input field and show back the page to the user.
- Those are convert to html tag only bz the browser don't know the spring tag, but to read the data from the request scope we are using those tag, those are intern classes only, those are executed and place in terms of html tag.
- When we are working with simple form controller we don't need to use different controller for the source and the target page. We can use the simple form controller for both the pages, but the problem is for to render the source page , we don't need to read the data from the request, bz the source page is the static page, in this page we are going to fill the data, this page will contain only fixed input control. We don't need the request wrapping as we don't send the data. We are accessing this page entering url or by clicking the hyperlink , so data will not send.
- So whenever dispatcher servlet identify the controller it call handle request method on our controller class bz it treat every controller as Controller interface type. As this method is not there in our class , so it goes to super class simple form controller which contain this method. Inside this method simple form controller try to find out on which request method the request is coming. As we are sending get request bz 1st time we are typing the url . It identify the get request.
- Then it create the empty command object and put it in request scope with name as command by default. As we are using spring form tag, by default it try to find the command object with name command in the request scope. If command object is not there it throw the exception. That's why simple controller create one empty command object with attribute value as default value and place in the request scope. And give one form view as view name and return the ModelAndView object back to the Dispatcher servlet . Then servlet render the source page with empty input field. Bz the command object is empty in the scope.

- And whenever we put the data in the jsp page and submit , then we need to read the data from the request , to render the target page, so we need request wrapping and data validation, as to render both the source and target page we are using same controller , then how did it understand for which request it has to do what.
- (That's why simple form controller works on 2 phase, one is initial phase and the other one is post back phase. In the initial phase it has to render the source page or the static page, and in the post back phase it has to render the target page.
- Whenever the request comes to the simple form controller it check the request method (get or post) of the incoming request, to identify the phase, so it goes to the request and check the method, if the request method is get request, then it understand that this is a request to perform the initial phase. So after receiving the request it doesn't do anything , it simply return the logical view name of the jsp page that has to be render for that request. So that that is how the source page will be render.)
- Again when we send the request by submitting the form then dispatcher servlet delegate request to this controller , then it check the request method, as it is a post request bz the data is coming as part of the request , it understand it has to process the request, so the onSubmit() will be called on our controller, here it perform the request wrapping by reading data from request and putting in the command object, and return the logical view name of the target page.
- Then it create one empty BindException object , and it wrap the request parameter into the command object , then it call our validator class to validate the data, the logic we are writing inside our class. If any validation error occur as we are putting all the error inside the error object, after validating it check wheather the error object is empty or not, if empty then no data validation error and it call our onSubmit() , but if errors are there it put the command object with the error object in the request scope , and show back the same page, so the spring form tag read the data from the scope , and show the error information , by getting the key from the scope and we are giving th value for that key in properties file.
- If we configure the errors description as part of the validator then we cannot display errors description in their native language that means internationalization not possible hence we should not populate the errors in the java code rather we need to display using properties file with the help of ResourceBundleMessageSource.

Java Bean validation

- While we are working In application we have to validate the data that the user is submitting from the jsp page , so we are using validator as part of the application which is spring specific , but java provided one more api called java bean validation , to validate the data.
- We have to create the object of validator

  ValidatorFactory factory=Validation.DefaultValidatorFactory();

  Validator validator=factory.getValidator();

  Set<ConstraintViolation> constraints=validator.validate();

  With this the validator goes to the attribute read the annotation, and validate the data , it is the normal flow, if the annotation is not sufficient we can write our own annotation, like

  @target(field)---this is telling where you want write this annotation at attribute means field

  @Retention(RunTime)---this is telling at the runtime means jvm also read this annotation

  @Documentation

  @Constraint(validatedby=PhoneNumberValidator. class)

  @interface PhoneNumber{

  Public String message default ""{}---this is method is for writing the message for the validation,

  Public Class<?> groups default {}----this is for if you want to use this for nested annotation,

  Public Class<?> payload default {}----this 3 are acting as attribute for the annotation

  }
- We have to write the validation logic inside phone number validator class, and the @phonenumber annotation at the attribute level, so that when the validator read the annotation , it goes to the phone validator class, and validate the logic , if fail it goes to the annotation message attribute and read th message , and show if we did not wrote any message then it goes to the annotation class and read the default message for message attribute.
- So we have to write the annotation at the attribute of the form class.so how the annotation are working. And in the controller handler method we have to write one annotation called @valid, so when adapter see this annotation it automatically call the java validation annotation which we wrote at attribute level , or if provide any validator class

also it automatically call our validator class, we don't need to call our validator class.so how it calling,

- Whenever adapter calling our method by seeing model attribute annotation it identify that it has to bind the request data into the form object , so while binding if any validation failure happen then it add to the binding result means errors, then it see we are writing @valid annotation , so it has to valid our java validation annotation also.
- But for validating the annotation it needs validator, and to create validator it needs Validator factory, so we have to provide the validator factory to the adapter, by default spring integrate with hibernate and use hibernate provided factory, so it create the validator factory and get the validator from the factory, and it call validate method, so in result it is getting Set<ConstraintViolation> ,
- But , our controller method need binding result , bz in spring form tag we are writing form:errors, which is by default looking for errors in the request scope, So handler adapter iterate over the constraint violation get the key and place them in the binding result and pass the errors.
- Then after binding the constraint into the errors, it check wheather you wrote any validator or not in the init binder (we have to register our validator class into the init binder so that we don't need to call adapter automatically call our validator) , if provide then it validate and put the errors in the binding result then only it call our handler method by passing form object and errors and model.

SPRING-SECURITY

- When we are working with security 1st of all we have to have a login page, so that we can collect the identity of the user , means user name and password. suppose we have a jsp page called login.jsp and a servlet who receive the request from the jsp page called login handler. After collecting the user name we have to verify the identity, so we have to store the user name and password somewhere else , so authentication store comes into picture.
- Authentication store means it's a type of storage in which we can store the user identity data. It can be file, data-base, or a LDAP server. We can store in the file system but there is a problem, there is no security in file, bz any one can read and modify or corrupt the file easily, so we don't have security in storing in the file system.
- So we can go for data base, in which we have to store the username, password and the role of the user. But the people who has the access to the database of the application can stole the data, here also there is the problem, so we have to encrypt the password, and store. Here the question is if you encrypt and store how you validate the data,

always we have to encrypt the data In the program and we have to compare with the encrypt data of the data base, don't get the encrypt data from the data base and decrypt it and compare. If you allow to do this then also any one can decrypt the data and see the data, if he is a technical person. So don't allow to any one to decrypt, you itself encrypt and compare with the encrypt data of the data base. Generally people are using salted encryption algorithm in encrypting the data.

- Instead we can go for ldap server also which is more secure that any other, we can say we can achieve full security if we are using ldap server. Bz we just need to give the username and password , and itself he is using his own algorithm in encrypting the data , means here we can only give the data to the ldap, we cant get the data from him. So when user submit its identity then we just have to give the clear text of data to ldap , bz we don't know in which way he store the data, so ldap itself capable of comparing the data with his stored data, in its own way. So we can achieve at most security if we go for ldap server.
- So lets think we have one of the authentication store in our application, so we have to talk to the authentication store, and if we are writing the logic to talk to the authentication store inside our class then our class, will be tightly coupled with the one of the authentication store that we are using, then if we want to change then we have to modify the logic inside our class. So don't write the logic in talking to ldap server in your class, so authentication provider comes into picture, write the logic inside the provider to talk to authentication store. And your class need to talk with the authentication provider. So if we change the authentication store from one to another we don't need to modify the code , bz we are not talking to the authentication store directly.
- Generally these authentication provider are interface driven, bz if we have multiple security authentication mechanism , then we have to have multiple authentication provider implementation, means if we are using username and password base authentication then we can use password authentication provider, similarly if we have another authentication mechanism then we can write one more implementation , so that we can change from one to another authentication provider easily.
- So we have finally authentication provider, so our class has to talk to the provider, but if he is talking to our class. Then again some problem will arise, as we have flexibility in working with multiple security mechanism, and we are talking with one of the mechanism authentication provider, so if we want change from one authentication mechanism to another then we have to modify our class code, bz we are directly talking to the provider. So here authentication manager comes into picture.
- We can write lot of functionality to the authentication manager, lets talk in incremental way, so we have to write the

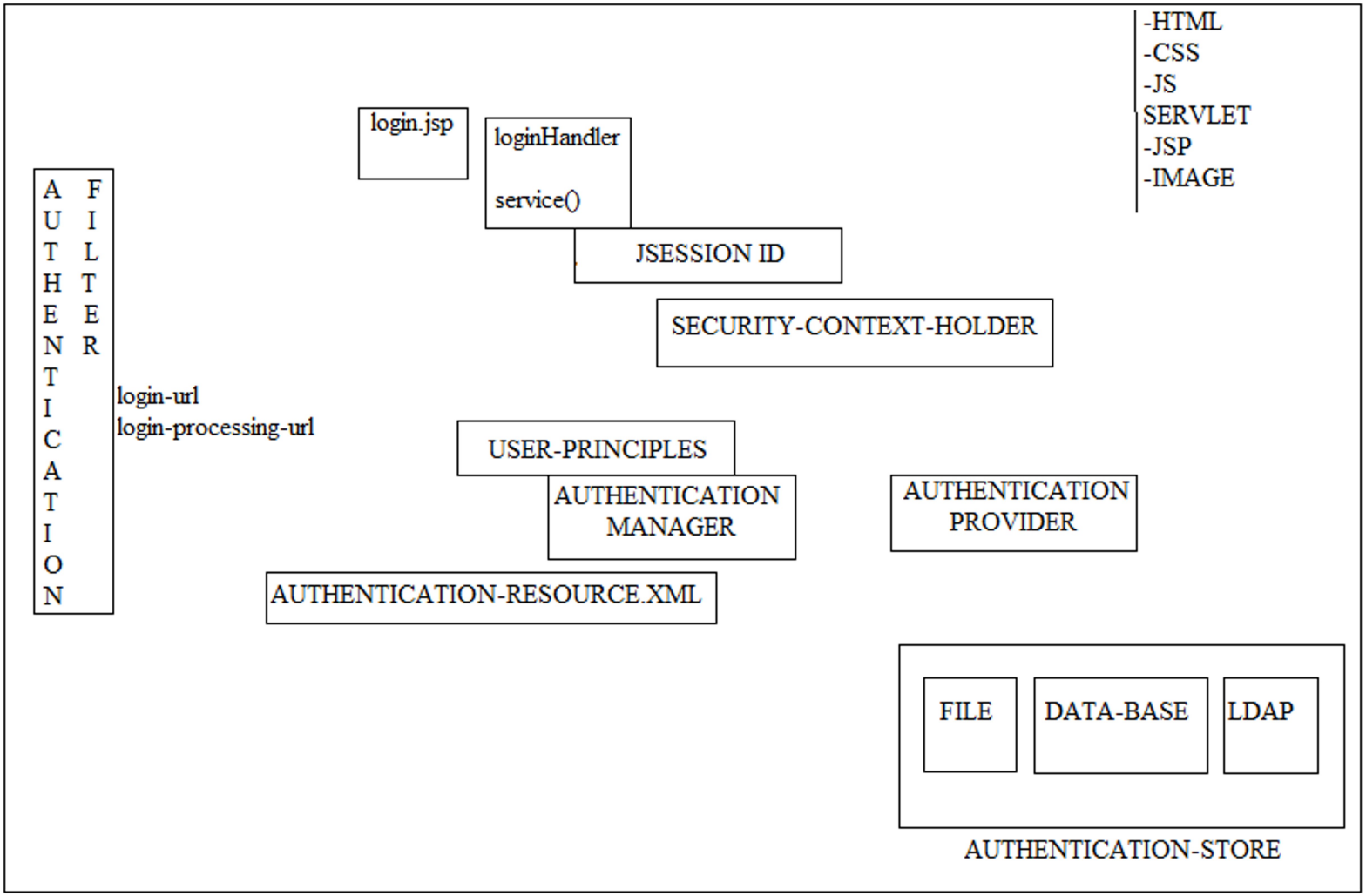
logic for talking to the provider inside the manager. we have to write the comparison logic for validating the username and password in the manager, So when the user login into our application, the user is login by submitting username and password in the jsp page. And then the request comes into our login handler , and the login handler goes to authentication manager and give the username and password to validate .

- So authentication manager goes to provider and provider goes to authentication store and get the user data from the store give to the provider, and provider give the data to the manager ,and manager is validating the data , if the user data is wrong then we can throw the exception or we can redirect the same page to the user to again provide the username and the password, but if the data is correct we have to give the access to the user to our resources.
- So again when the user send the 2nd request we have to again say to the user login again, bz the user is logged in 1st time and we are not storing the logged in data, so 2nd time we don't have the data, so we have to again say to login for our security requirement, which is a bad way , the user may not feel comfortable in using our application, bz we are forcing him to again and again for login.
- So when the user 1st time logged in into our application we have to store the login information some where else , so that we don't need to say again and again for login to the user, for the consecutive request. So we have to create one http session for the user and we have to store the login information of the user, so that we can check wheather the user already login or not.
- So we have to store the user login data in the session and we can check wheather the user is logged in or not by going to the session. And we have to create one jSessionId for the user who logged in , and we give that id to the user, so that when 2nd time he is accessing the application , then he has to send the request along with jSessionId for the consecutive request, so we can know wheather the user is logged in or not.
- But if we do this only then when the user send the request 2nd time we can identify wheather the user is logged in or not, but we cant identify who is that user, bz 2nd time the user is not sending his identity, so we cant know who is the user and what is the role of the user, so when the user is logged in we create one http session with jSessionId and we have to store the user details in some where else. The authentication manager will give us the user details, bz he is only verifying that wheather the user is valid user or not. provider giving the user details by going to the authentication store. So we can store the user details in the http session. Of the user which we created.
- But if we want to know how many user are currently logged in user or we may have the requirement in collecting all the information about the currently logged in user and we may want to stop giving all the user to access our

application to save from hacking issue for some time. If we are storing all those information in the session of the user we cant get all the user session data , such mechanism is not there.

- So security context holder comes into picture, so after creating the session for the user, store the session with the jSessionId in the Security context holder.so that we can go to the security context holder and we can check wheather the user is logged in or not, and we have a global memory location where we can store all the user information who are currently logged in.
- But the problem is the user can by pass from the login page and directly can try to access our resources without login. So It is our responsibility to write the logic in all the resources of our application to check wheather the user is logged in or not in going to the security context holder. Before granting the access to the user to access our application. But here the problem is the logic will be duplicated across the resources, and we cant write this logic in some static content also, means we can write the logic in dynamic resource like servlet or jsp, but static content like html, css, or image file we cant.
- So here authentication filter comes into picture. Write the logic in verifying the user is logged in or not inside the authentication filter. And all the request should go through the filter only, then only we can achieve the security. And another thing is, we don't need login for accessing all the resources of our application, bz some of the resources can be accessible without logged in also. Like home page, in amazon the user can see the product without login, if he want to buy some product then only the user has to login, for all the component user don't need to login.
- But the authentication filter don't know which resource are authenticated resource means protected resource or which are not, so we have to give the information about our resource of our application which are authenticated which are not to the filter by writing one configuration file. Inside we have to write which are authenticated resource which are not.
- So whenever the filter receive the request(the request comes into our application means it receive the request bz it is configure with url as /* that we have to configure) so filter has to check wheather the user is a logged in user or not , so it has to go to the security context holder and check wheather the user is already logged in or not. If he logged in then grant the access to the resource page. If he is not logged in , then it has to heck wheather the request url is login url(means the login page) or login processing url(means the login handler servlet url) , if it is one of this url then it forward to the corresponding component, but if it is not one of them then it has to identify wheather the resource for which the request is coming is authenticated resource on not.

- So it has to go to the authentication manager and should give the request and ask him for telling wheather the resource is protected or not, so it is the responsibility of the manager to go to the configuration file and verify , if the resource is non-authenticated then filter will redirect to the resource page directly , but if the resource is protected resource then the filter redirect the request to the login page and ask him for login. And after logged in it should redirect the user to the target resource.
- **Full Flow**----- when the request is coming into our application, authentication filter receive the request, and it go to the security context holder and check wheather the user is already logged in or not , as 1st time he is not logged in so filter then check wheather the resource is protected resource or not . So it go to authentication manager and ask him for giving information about the resource.
- So manager goes to the configuration file and check , if the resource is non protected then he can directly can forward to the resource, but if the resource is protected and the user is not logged in then the filter will redirected the request to the login page.
- The user is provide the username and password in the login page, and the request comes to login handler , handler get the username and password and go to the authentication manager and give the username and password and ask him to validate. Then authentication manager goes to the authentication provider then the provider goes to authentication store. And store return the details of the user back to the provider, and provider gives back to the authentication manager,
- Authentication manager compare wheather the username and password is valid or not in comparing the system stored data with the user submitted data, if it doesn't match then it throw the exception, if it match then the manager will return the user details or it can be called as user principles object back to the login handler , then the login hander create one http session with one jSessionId and store the session object in the security context holder, with key as jSessionId and value as session object. And give the jSessionId to the user .
- So that when he send again the request to our application within the session time then the request will be receive by the filter and he can go to the security context holder and get the current user information, as he is already logged in filter grant the user to access the resource. That's how the spring security flow is going to work.



Spring-Security

- To work with security 1st of all we have to configure the authentication filter. Inside the ioc container. Bz this class has to talk to the service class, and the service class has to talk to the dao class, it should not directly talk to the dao. That's why we have to configure the filter as a bean inside the ioc container. Bz the service and the dao are the bean inside the ioc container, then only we can inject service into the filter.

- But if we place the filter inside the ioc container the servlet container cant call the filter, bz to work with security for every request we have to call 1st the filter. But filter can be call by only servlet container. So we have to configure the filter in the web.xml . But if we place in the web.xml then filter cant call the service and we cant inject service into the filter.
- So the solution is for every request filter should create one ioc container. Then filter can call the service from the ioc. But if we do this , then memory management issue comes, bz we are creating unnecessarily memory. So we have to do one thing that if we place the ioc container in the servlet context then the filter can go to the servlet context and get the reference of the ioc container , but for every servlet or filter we have to write the logic for getting the ioc container from the servlet context. And here we are doing dependency pulling.
- So spring said place your original filter class inside the ioc container with one bean id= springSecurityFilterChain and give the ioc container to the ContextLoaderListener , so that listener after creating ioc container it will place the ioc container inside the servlet context. But if we do this servlet container cant call the filter.
- So spring provide one class called DelegatingFilterProxy which is a proxy class of our original class, and tell to us that you configure this proxy filter inside the web.xml with name as springSecurityFilterChain , and place the original filter inside the ioc container with id same as the name of the filter inside the web.xml springSecurityFilterChain, and the url of the filter as /*,
- So when the request comes servlet container will receive the request and call the delegating proxy filter inside the web.xml. This proxy class has the logic like, 1st it will pick his name and go to the servlet context and get the reference of the ioc container and search for the bean definition whose id match with his name. If it find then it call that filter. And that filter call our service class , that s how we have to do.

Spring-Restful

- When we are working with spring with restful we need to make our resource class as bean inside the ioc container , bz as we are go for spring, then if the resource class has to talk with the service class then the resource has to go inside the ioc container and get the service which is already a bean inside the ioc container.
- If we are not making our resource as bean inside the ioc container then we have to get manually the service class from ioc by getting the ioc container. So make the resource class as bean inside the ioc container and inject the

service class into the resource so that all the dependency management will be done by the ioc container.

- But if we make the resource class as bean inside the ioc container then upon receiving the request by the servlet container of the jersy runtime it will looks only the resource class inside the application class as singletons , it will not look for ioc container so, jersy provide one servlet for the integration , as spring servlet which will look for the ioc container but this has integration on the restful service of version 1.0, it has not provide integration with latest restful version 2.0.
- So make the resource as bean inside the ioc container, and inject the service into the resource so it will talk to the service and the service will talk to the dao and fulfil the request. and give the ioc container to the context loader listener , so that it will add to the servlet context, and then we have to manually get the ioc container from the servlet context .
- So inside the application class we have to go to the servlet context and get the ioc container from the servlet context , and get the resource bean from the ioc container and add to the singleton so that upon receiving the request the jersy runtime will go to the application class which are added in the singletons or class , and get the resource and call the resource which will talk to the service and the service will talk to the dao and fulfill the request.

```
@ApplicationPath("/api")
public class SpringRestApplication extends Application {
	private Set<Object> singletons;

	public SpringRestApplication(@Context ServletContext servletContext) {
		List<String> resources = null;
		ApplicationContext context = null;

		resources = (List<String>) servletContext.getAttribute("resources");
		singletons = new HashSet<>();

		context = WebApplicationContextUtils.getRequiredWebApplicationContext(servletContext);
		for (String beanName : resources) {
			singletons.add(context.getBean(beanName));
		}
	}

	@Override
	public Set<Object> getSingletons() {
		return singletons;
	}
}
```

- Spring-Rest---

```
                    in older version @ResponseBody
@RestController
@RequestMapping("/wallet")
public class PayTMWalletController {
        @GetMapping("/{mobileNo}")
        public float getBalance(@PathVariable("mobileNo") String mobileNo) {
                System.out.println("mobile : " + mobileNo);
                return 33.34f;
        }

        @PostMapping(consumes = { "application/json" }, produces = { "application/json" })
        public ResponseEntity<Receipt> addBalance(@RequestBody Recharge recharge) {
                Receipt receipt = null;
                ResponseEntity<Receipt> responseEntity = null;

                receipt = new Receipt();
                receipt.setMobileNo(recharge.getMobileNo());
                receipt.setBalance(100 + recharge.getAmount());
                responseEntity = new ResponseEntity<>(receipt, HttpStatus.CREATED);
                return responseEntity;
        }
}
```

We can expose our controller as resource if we are working with spring restful , but the controller are returning logical view name, but we want the resource has to return the output so spring provide annotation ,By looking the annotation as @RestController or @ResponseBody(if we are working with older version and the controller is annotated with @Controller and response body but in latest version spring provide rest controller which is combination of controller and response body) it will not return the logical view name , whatever the output comes it will write into the output stream,

@RequestBody--if we want to receive the Request body as input , in restful service if we don't annotate the resource method parameter with any annotation then it will pass the request body, but in spring they provide the annotation for it as request body,

ResponseEntity--it is equivalent with Response in restful Service

Spring-Boot

- Spring boot doesn't provide any functional requirement , it is only meant for providing non-functional requirement like configuration , we don't need to write java configuration classes , spring boot provide auto configuration classes, for configuring all the spring internal component like handler mapping , and all these things.
- So in pom.xml we don't need to add the spring dependency , instead of this dependency boot provide starter dependency , which is a pom.xml file which will contain all the spring jar dependency as transitive dependency. Similarly spring provide 41 starter dependency which internally has spring dependency as transitive dependency. Lets take one example if we take spring starter dependency as 1.65 version which internally contain all the spring 4version jar as transitive dependency , if we take 2.0 then it contain all the spring 5.0 jar version as transitive dependency .
- And instead of this we need to configure maven war plugin, maven compiler plugin, and we have to add all the spring starter dependency with same version no, bz those are contain transitive dependency as spring jar, so spring provide one parent pom.xml file which is present in central repository. This parent pom.xml is a pom file of packaging as pom (if you have some common dependency that you have to add in every project , then write one pom.xml with packaging structure as pom, and import this pom file in every pom.xml file so that it will inherit to all the pom file, so we don't need to add those dependency in all pom file). Which contain all the 41 spring starter dependency with war plugin and all this, so if we add this parent in the pom.xml we don't need to add the dependency.
- We just need to write the spring starter which we are going to import and all these things will automatically import from the parent , we don't need to even tell the version of the dependency also.
- And in the test class we need to write one annotation @SpringBootApplication which is a combination of 3 annotation @Configuration, @ComponentScan, @EnableAutoConfiguration , enable auto configuration is meant for enabling auto configuration classes , like if we want data source or JdbcTemplate we don't need to write configuration classes, boot provide auto configuration classes for them.
- But the auto configuration classes are created by default means if we want data source then it create data source with default db , but we want the data source to be created with our own data base then we need to write those information in one properties file called application.properties. The auto configuration classes are read from this properties file and create the component which we want, what ever you want the configuration classes are going to create with your ow write it inside the properties file.

ApplicationContext context = SpringApplication.run(SDJPATest.class);--this spring application has the logic for creating the ioc container with the test class here we are making the test class as configuration classes.

- Spring boot has also provide one concept called actuator, which will helps us in getting the health status of an application, or if we want to monitor the logins or if want to know about the application, or which are the component that is used in our application , we just need to add one dependency and we have to add in the application.properties file. Spring will auto configure all those component.
- Spring boot maven plugin will create one jar called uber-jar which is a special jar which will have one jar launcher class loader . Which include all the dependency jar inside the manifest file, that's why we are making our boot application as normal jar , we don't need to make the application as war. We can make the application as jar and we can deploy this jar, which is contain their own application server on which the application is deploying.