SPRING FRAMEWORK #11

Date of the Session: ___/___/___ **Time of the Session:** _____to_____

Prerequisite:

- General idea on Spring Framework
- Modules of Spring Framework

Pre-Lab Task:

1. What is a spring?

Spring framework is an application framework & inversion Control Container for the java Platform. the frameworks are features can be used by any java application but there are extensions for building web application JEE Platform

2. Name some of the important spring Modules?

Spring Content - For dependency injection
Spring AOP - For aspect Oriented Programming
Spring DAO - For database operations using DAOP often
SPring JDBC - For JDBC & data source support
Spring ORM - For ORM tools Support Such as hebernate

3. What are the tasks performed by IOC container?

The IOC Containes is responsible to instantiate, Configure & assemble objects. The Ioc Containes get Information from Xml file & works accordingly. The main tasks Performed by Ioc Containes are to Instantiate the application class.

4. How many types of IOC containers are there? Explain them?

SPring bean factory Containes
Spring application Context Containes

In Lab Task:

1. Geetha is creating a website which displays the marks of student from two different java classes. So, define two different java beans and input attributes in Student.java like studentId, studentName and Marks.java which containing clear segregation of marks scored in each subject. Use Spring FrameWork to display all the details of each student to the browser.

```
Package myPackage;
  Public class Student{.
        Private int Sid;
        Private String SName;
        Private int Mark;
  Public void SetStId (Int Stid){.
            this.StId = Stid ;}.
  Public void SetStName (String Name){.
            this.StName=St Name;}
  Public void set Mark (Mark mark) {.
            this.mark =mark ;}}.
  Public Mark get Mark() {.
            Return Mark ;}.
  Public void Set Mark (Mark mark) {.
            this.mark =mark;
    }
  }.
Public  Class Mark{.
  Public String Maths;
  Public String Physics ;
  Public String Chemistry;.
  Public String get Maths() {.
            Return Maths ;}.
```

```
Public String getPhysics() {
        return Physics; }
Public String setPhysics(String Physics) {
        this.Physics = Physics;
    }
Public String setChemistry(String chemistry);
        this.chemistry = chemistry
        }
    }
```

Test.java :-

```
Package myPackage;
import org.Spring framework.beans factory.BeanFactory.
import org.SPring framework.Core.io.*;
Public class Test {
    Public Static void main(String[] args)
    {
    Resource resource = new classPath resource("application
                                                context.xml");
    BeanFactory f = new xmlBeanFactory(resource);
    Student s = (Student) Factory.getBean("st")
System.out.println("s Id" + s.getId());
System.out.println("st mark");
System.out.println("maths" + s.getmark().getmaths());
System.out.println("Physics" + s.getmark().getPhysics());
```

App.Content.xml :-

```
<?xml version="1.0" encoding="utf-8"?>
<beans>
<bean id="mark" class="mark">
<Property name="maths" value="50"></Property>
```

Writing space for the Problem:(For Student's use only)

```
<Property name = "Physics" value = "48" ></Property>
<Property name = "chemistry" value = "45" ></Property
</bean>
<bean id = "st" class = "Student" >
 <Property name = "sid" value = "123567" >
</Property>
<Property name = "stName" value = "Sri" >
</Property>
</bean>
</beans>
```

2. Mr.Deepak is very keen and interested to overcome tightcoupling in java so he used Spring Framework with Dependency injection mechanism, he created a class traveler containing an interface vehicle and a method startJourney calling a move method when a journey starts. He creates another two classes bike and car which implements the vehicle class, creates a object for vehicle when a journey starts. To know which vehicle is being used he is calling the method move with vehicle object so that the message will display which is present in implemented classes.

Journey.java:

```
Public interface Journey{
        void StartJourney();}
```

Travel.java:

```
Public class Travel implements Journey{
  Private vehicle v;
  Public void setV(Vehicle V) {
  this.v=v;}
  Public void Start journey(){S.o.println("journey starts....");
  V.more();
  }
}
```

Car.java:

```
Package myPackage;
  Public class Car implements vehicle{
  Private String FuelType;
  Private int maxSpeed;
  Public String getfuelType(){
        return fuelType;}
  Public void setfuelType(String FuelType){
        this.fuelType = FuelType;}
  Public void setmaxSpeed(int maxSpeed){
        this.maxSpeed = maxSpeed;}
  Public void move(){
        System.out.println("fuelType:"+FuelType);
        System.out.println("maxSpeed", maxSpeed);
        System.out.println("Bus started)";}}}
```

Config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans>
<bean id="car" class="Car">
<Property name="fuel type" value="diesel"></Property>
<Property name="maxSpeed" value="100">
<bean id="bus" class="Bus">
<Property name="maxSpeed" value="80"></Property>
</bean>
</beans>
```

Config 2. xml

```
<? xml version="1.0" encoding="UTF-8"?>
<beans>
<bean id="travel" class="Travel"
<Property name="vehicle">
<ref-Parent="bus"/>
</Property>
</bean>
</beans>
```

Logic . java :

```
Public State void main (String[] args).
{
Resource resource = new classpath Resource ("Config 1.xml");
        BEan Factory = new xml Bean Factory (resource);
Resource res =new class Path Resource ("Config 2.xml");
        Bean Factory = new xml Bean Factory (res);
    object o = res. get Bean ("travel");
    Journey j = (Journey) o;
```

Post Lab Task:

1. Ms.Varsha wants to develop an application using Spring FrameWork to display all the details of each employee to the browser. It displays the address of employee from two different java class, so define different java beans and input attributes in Employee.java like empId, empName and Address.java which contains clear address of each person.

```
Package myPackage;
Public class Employee{
        Private int empId;
        Private String empName;
        Private Address adr;
Public int getEmpId(){
        return empId;}
Public void setEmpId(int empId){
        this.empId=empId;}
Public String getEmpName(){
        return empName;}
Public void setEmpName(String empName){
        this.empName=empName;}
Public Address getAddr(){
        return addr;}
Public void setAddr(Address addr){
        this.addr=addr;
    }
}
```

Address.java :

```
Package myPackage;
Public class Address{
        Private String street;
        Private String city;
        Private String state;
Public String getStreet(){
        return street;}
```

```
Public void set Street (Street Street ){.
        this.Street =Street; }.
Public String get City() { return city; }
Public void set City (String City ) { this.City = City ; }
Public String get State () { return State ; }
Public void set State (String State ) {
        this.State = State; } }.
```

Test.java :

```
Package myPackage;
import org.Spring framework.beans.*;
import org.SPring Framework.factory.*

Public class Tests {.
    Public static void main ( String [] args)
    {
    Resource resource = new classpath Resource ("application Context.
                                                    xml");

    Bean factory = new xml Bean factory (resource);
    employee e = (employee) factory.get Bean ("emp");
  System.out.Println ("empId" e.get empId() );
  System.out.Println ("empName" + e.get empName() );
  System.out.Println ("emp Address");
  System.out.Println ("Street" + e.get addr() .get Street() );
  systemout.Println ("Street " + e.get addr() .get{ Street() );

  }
}
```