**Spring Assignment**

**1. Create an address class with the following attributes : street ,city , zip ,state, country, create an customer class with the following attribute – customerid, customerName, customerContact, CustomerAddress.**

**Inject the Addresss bean into Customer bean using settervinjection.**

**Create a Test class with main() method, get Customer bean from ApplicationContext Also and print details of customer .**

**Also Write the Junit Test cases for above program.**

**-modify the above application and inject the bean using constructor injection.**

**- Use XML based configuration**

**Address.java**

**package** Assignment1;

**public** **class** Address

{

**private** String Street;

**private** String City;

**private** String State;

**private** String zip;

**private** String country;

**public** String getStreet() {

**return** Street;

}

**public** **void** setStreet(String street) {

Street = street;

}

**public** String getCity() {

**return** City;

}

**public** **void** setCity(String city) {

City = city;

}

**public** String getState() {

**return** State;

}

**public** **void** setState(String state) {

State = state;

}

**public** String getZip() {

**return** zip;

}

**public** **void** setZip(String zip) {

**this**.zip = zip;

}

**public** String getCountry() {

**return** country;

}

**public** **void** setCountry(String country) {

**this**.country = country;

}

@Override

**public** String toString() {

**return** "Address [Street=" + Street + ", City=" + City + ", State=" + State + ", zip=" + zip + ", country="

+ country + "]";

}

}

**Customer.java**

**package** Assignment1;

**public** **class** Address

{

**private** String Street;

**private** String City;

**private** String State;

**private** String zip;

**private** String country;

**public** String getStreet() {

**return** Street;

}

**public** **void** setStreet(String street) {

Street = street;

}

**public** String getCity() {

**return** City;

}

**public** **void** setCity(String city) {

City = city;

}

**public** String getState() {

**return** State;

}

**public** **void** setState(String state) {

State = state;

}

**public** String getZip() {

**return** zip;

}

**public** **void** setZip(String zip) {

**this**.zip = zip;

}

**public** String getCountry() {

**return** country;

}

**public** **void** setCountry(String country) {

**this**.country = country;

}

@Override

**public** String toString() {

**return** "Address [Street=" + Street + ", City=" + City + ", State=" + State + ", zip=" + zip + ", country="

+ country + "]";

}

}

**Test.java**

package Assignment1;

import org.springframework.beans.factory.BeanFactory;

import org.springframework.beans.factory.xml.XmlBeanFactory;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.core.io.ClassPathResource;

import org.springframework.core.io.Resource;

public class Test {

public static void main(String[] args)

{

ApplicationContext context = new ClassPathXmlApplicationContext("Spring1.xml");

Customer customer= (Customer) context.getBean("Cust");

customer.Display\_CustomerInfo();

}

}

**Xml file :**

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

<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN 2.0//EN" "http://www.springframework.org/dtd/spring-beans-2.0.dtd">

<beans>

<!-- <bean id="customer" class="Assignment1.Customer">

<property name="CustomerId" ref="CustomerId"> </property>

<property name="CustomerName" ref="CustomerName"> </property>

<property name="CustomerContact" ref="CustomerContact"> </property>

<property name="CustomerAddress" ref="CustomerAddress"> </property>

</bean>

-->

<bean id=*"Address"* class=*"Assignment1.Address"*>

<property name=*"Street"* value=*"Tikrapara"*></property>

<property name=*"City"* value=*"Raipur"*></property>

<property name=*"State"* value=*"Chhattisgarh"*></property>

<property name=*"zip"* value=*"492001"*></property>

<property name=*"country"* value=*"India"*></property>

</bean>

<bean id=*"Cust"* class=*"Assignment1.Customer"*>

<property name=*"CustomerId"* value=*"1"*></property>

<property name=*"CustomerName"* value=*"Palak"*></property>

<property name=*"CustomerContact"* value=*"700"*></property>

<property name=*"CustomerAddress"* ref=*"Address"*></property>

</bean>

</beans>

**2.Example of injecting collections….**

**Question.java**

**package Assignment2;**

**import java.util.Iterator;**

**import java.util.List;**

**public class Quesion**

**{**

**private int QuesionId;**

**private String Quesion;**

**private List<String> Answer;**

**public int getQuesionId() {**

**return QuesionId;**

**}**

**public void setQuesionId(int quesionId) {**

**QuesionId = quesionId;**

**}**

**public String getQuesion() {**

**return Quesion;**

**}**

**public void setQuesion(String quesion) {**

**Quesion = quesion;**

**}**

**public List<String> getAnswer() {**

**return Answer;**

**}**

**public void setAnswer(List<String> answer) {**

**Answer = answer;**

**}**

**public void Print()**

**{**

**System.out.println("Quesion Id = " +QuesionId);**

**System.out.println("Quesion is = " +Quesion);**

**System.out.println(" Answers are :");**

**Iterator<String> i= Answer.iterator();**

**while(i.hasNext())**

**{**

**System.out.println(i.next());**

**}**

**}**

**}**

**Test.java**

package Assignment2;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Test {

public static void main(String[] args) {

ApplicationContext context = new ClassPathXmlApplicationContext("Spring2.xml");

Quesion quesion= (Quesion) context.getBean("Ques");

quesion.Print();

}

}

**Spring.xml**

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

<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN 2.0//EN" "http://www.springframework.org/dtd/spring-beans-2.0.dtd">

<beans>

<bean id=*"Ques"* class=*"Assignment2.Quesion"*>

<property name=*"QuesionId"* value=*"01"*></property>

<property name=*"Quesion"* value=*"What is Spring Framework?"*></property>

<property name=*"Answer"*>

<list>

<value>Spring is an open source framework. </value>

<value>It is an application framework and inversion of control container.</value>

<value>Framework for building enterprise java application.</value>

</list>

</property>

</bean>

</beans>

**4. Example on @controller @service @repository @autowired @configuration @bean**

@Bean

**package** spring\_ex4;

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

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

//@Configuration

**public** **class** ApplicationConfiguration {

@Bean(name="demoService")

**public** DemoManager helloWorld()

{

**return** **new** DemoManagerImpl();

}

}

**package** spring\_ex4;

**public** **interface** DemoManager {

**public** String getServiceName();

}

**package** spring\_ex4;

**public** **class** DemoManagerImpl **implements** DemoManager

{

**public** String getServiceName()

{

**return** "Hello!!!!";

}

}

**package** spring\_ex4;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.annotation.AnnotationConfigApplicationContext;

**public** **class** VerifySpringCoreFeature {

**public** **static** **void** main(String[] args)

{

ApplicationContext context = **new** AnnotationConfigApplicationContext(ApplicationConfiguration.**class**);

DemoManager obj = (DemoManager) context.getBean("demoService");

System.***out***.println( obj.getServiceName() );

}

}

Output:

Jan 25, 2022 2:01:42 PM org.springframework.context.support.AbstractApplicationContext prepareRefresh

INFO: Refreshing org.springframework.context.annotation.AnnotationConfigApplicationContext@20322d26: startup date [Tue Jan 25 14:01:42 IST 2022]; root of context hierarchy

Addition of first and second = 4

Jan 25, 2022 2:01:43 PM org.springframework.context.support.AbstractApplicationContext doClose

INFO: Closing org.springframework.context.annotation.AnnotationConfigApplicationContext@20322d26: startup date [Tue Jan 25 14:01:42 IST 2022]; root of context hierarchy

@contoller

**package** spring\_exp4;

**import** org.springframework.context.annotation.AnnotationConfigApplicationContext;

**public** **class** SpringMainClass {

**public** **static** **void** main(String[] args) {

AnnotationConfigApplicationContext context = **new** AnnotationConfigApplicationContext();

context.scan("spring\_exp4");

context.refresh();

MathController ms = context.getBean(MathController.**class**);

**int** result = ms.add(2, 2);

System.***out***.println("Addition of first and second = " + result);

context.close();

}

}

package spring\_exp4;

**package** spring\_exp4;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.stereotype.Service;

//@Service("ms")

@Controller

**public** **class** MathController {

**public** **int** add(**int** x, **int** y) {

**return** x + y;

}

}

Output:

Jan 25, 2022 2:04:52 PM org.springframework.context.support.AbstractApplicationContext prepareRefresh

INFO: Refreshing org.springframework.context.annotation.AnnotationConfigApplicationContext@20322d26: startup date [Tue Jan 25 14:04:52 IST 2022]; root of context hierarchy

Addition of first and second = 4

Jan 25, 2022 2:04:52 PM org.springframework.context.support.AbstractApplicationContext doClose

INFO: Closing org.springframework.context.annotation.AnnotationConfigApplicationContext@20322d26: startup date [Tue Jan 25 14:04:52 IST 2022]; root of context hierarchy

@Service

package spring\_exp4;

import org.springframework.stereotype.Component;

import org.springframework.stereotype.Service;

@Service("ms")

//@Component

public class MathService {

public int add(int x, int y) {

return x + y;

}

} **package** spring\_exp4;

**import** org.springframework.context.annotation.AnnotationConfigApplicationContext;

**public** **class** SpringMainClass {

**public** **static** **void** main(String[] args) {

AnnotationConfigApplicationContext context = **new** AnnotationConfigApplicationContext();

context.scan("spring\_exp4");

context.refresh();

MathService ms = context.getBean(MathService.**class**);

**int** result = ms.add(2, 2);

System.***out***.println("Addition of first and second = " + result);

context.close();

}

}

Output:

Jan 25, 2022 2:01:42 PM org.springframework.context.support.AbstractApplicationContext prepareRefresh

INFO: Refreshing org.springframework.context.annotation.AnnotationConfigApplicationContext@20322d26: startup date [Tue Jan 25 14:01:42 IST 2022]; root of context hierarchy

Addition of first and second = 4

Jan 25, 2022 2:01:43 PM org.springframework.context.support.AbstractApplicationContext doClose

INFO: Closing org.springframework.context.annotation.AnnotationConfigApplicationContext@20322d26: startup date [Tue Jan 25 14:01:42 IST 2022]; root of context hierarchy

@Autowired

**package** maths\_example;

**import** org.springframework.beans.factory.BeanFactory;

// org.springframework.beans.factory.xml.XmlBeanFactory;

**import** org.springframework.context.ApplicationContext;

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

**import** org.springframework.stereotype.Repository;

//import org.springframework.core.io.FileSystemResource;

//@Repository

**public** **class** Mainbean {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

//BeanFactory factory= new XmlBeanFactory(new FileSystemResource("spring.xml"));

ApplicationContext context=**new** ClassPathXmlApplicationContext("spring.xml");

Shape shape=(Shape)context.getBean("circle");

shape.draw();

}

}

**package** maths\_example;

**public** **class** Triangle **implements** Shape {

**private** Point PointA;

**private** Point PointB;

**private** Point PointC;

**public** Point getPointA() {

**return** PointA;

}

**public** **void** setPointA(Point pointA) {

PointA = pointA;

}

**public** Point getPointB() {

**return** PointB;

}

**public** **void** setPointB(Point pointB) {

PointB = pointB;

}

**public** Point getPointC() {

**return** PointC;

}

**public** **void** setPointC(Point pointC) {

PointC = pointC;

}

**public** **void** draw()

{

System.***out***.println("Draw triangle");

System.***out***.println(getPointA().getX()+ " "+getPointA().getY());

System.***out***.println(getPointB().getX()+ " "+getPointB().getY());

System.***out***.println(getPointC().getX()+ " "+getPointC().getY());

}

} **package** maths\_example;

**public** **interface** Shape {

**public** **void** draw();

}

**package** maths\_example;

**public** **class** Point {

**private** **int** x;

**private** **int** y;

**public** **int** getX() {

**return** x;

}

**public** **void** setX(**int** x) {

**this**.x = x;

}

**public** **int** getY() {

**return** y;

}

**public** **void** setY(**int** y) {

**this**.y = y;

}

}

**import** org.springframework.beans.factory.annotation.Required;

**import** org.springframework.stereotype.Component;

**import** org.springframework.stereotype.Repository;

//@Component

//@Repository

**public** **class** Circle **implements** Shape {

**private** Point center;

**public** **void** draw()

{

System.***out***.println("draw circle");

System.***out***.println("circle point" +center.getX() +center.getY());

}

**public** Point getCenter() {

**return** center;

}

//@Autowired

**public** **void** setCenter(Point center) {

**this**.center = center;

}

}

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

<beans xmlns = "http://www.springframework.org/schema/beans"

xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation = "http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

<bean id = "triangle" class="maths\_example.Triangle">

<property name="PointA" ref="pointA"/>

<property name="PointB" ref="pointB"/>

<property name="PointC" ref="pointC"/>

</bean>

<bean id = "pointA" class="maths\_example.Point">

<property name="x" value="0"/>

<property name="y" value="10"/>

</bean>

<bean id = "pointB" class="maths\_example.Point">

<property name="x" value="10"/>

<property name="y" value="10"/>

</bean>

<bean id = "pointC" class="maths\_example.Point">

<property name="x" value="20"/>

<property name="y" value="10"/>

</bean>

<bean id = "center" class="maths\_example.Point">

<property name="x" value="20"/>

<property name="y" value="10"/>

</bean>

<bean id = "circle" class="maths\_example.Circle">

<!-- <property names="center" ref="pointA"/> -->

</bean>

<bean class="org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor">

</bean>

</beans>

Output:

Jan 25, 2022 3:22:45 PM org.springframework.context.support.AbstractApplicationContext prepareRefresh  
INFO: Refreshing org.springframework.context.support.ClassPathXmlApplicationContext@2038ae61: startup date [Tue Jan 25 15:22:45 IST 2022]; root of context hierarchy  
Jan 25, 2022 3:22:46 PM org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions  
INFO: Loading XML bean definitions from class path resource [spring4.xml]  
draw circle  
circle point2010

**5.write a java program to demonstrate @resource,@insert,and @required annotations.**

**Spring @Required Annotation**

Employee.java

**package** assignment5;

**import** org.springframework.beans.factory.annotation.Required;

**public** **class** employee {

**private** String name;

**private** String designation;

**private** String company;

@Required

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getName() {

**return** name;

}

@Required

**public** **void** setDesignation(String designation) {

**this**.designation = designation;

}

**public** String getDesignation() {

**return** designation;

}

**public** **void** setCompany(String company) {

**this**.company = company;

}

**public** String getCompany() {

**return** company;

}

@Override

**public** String toString() {

**return** "Employee [name=" + name + ", designation=" + designation + ", company=" + company + "]";

}

}

**Appmain.java**

package springrequiredannotation;

import **org.springframework.context.ApplicationContext;**

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

**import assignment5.employee;**

**public class AppMain {**

**@SuppressWarnings("resource")**

**public static void main(String[] args) {**

**ApplicationContext ac = new ClassPathXmlApplicationContext("required-annotation.xml");**

**employee emp = ac.getBean("myemployee", employee.class);**

**System.out.println(emp.toString());**

**}**

**}**

**Required-annotation:**

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

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="

http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">

<context:annotation-config />

<bean id="myemployee" class="com.spring.pojo.Employee">

<!-- Required property -->

<property name="name" value="Charlotte O' Neil" />

<!-- Required property -->

<property name="designation" value="Technical Leader" />

<property name="company" value="Test Ltd." />

</bean>

</beans>

**@resource annotation**

**Componay.java**

**package** com.spring.pojo;

**public** **class** Company {

**private** String name;

**private** String location;

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getLocation() {

**return** location;

}

**public** **void** setLocation(String location) {

**this**.location = location;

}

@Override

**public** String toString() {

**return** "Company [name=" + name + ", location=" + location + "]";

}

}

**Employee.java**

**package** com.spring.pojo;

**import** javax.annotation.Resource;

**public** **class** Employee {

**private** String id;

**private** String name;

@Resource(name="mycompany")

**private** Company company;

**public** String getId() {

**return** id;

}

**public** **void** setId(String id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** Company getCompany() {

**return** company;

}

**public** **void** setCompany(Company company) {

**this**.company = company;

}

@Override

**public** String toString() {

**return** "Employee [id=" + id + ", name=" + name + ", company=" + company.toString() + "]";

}

}

Appmain.java

**package** com.spring.util;

**import** org.springframework.context.ApplicationContext;

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

**import** com.spring.pojo.Employee;

**public** **class** AppMain {

@SuppressWarnings("resource")

**public** **static** **void** main(String[] args) {

ApplicationContext ac = **new** ClassPathXmlApplicationContext("resource-annotation.xml");

Employee emp = ac.getBean("myemployee", Employee.**class**);

System.***out***.println(emp.toString());

}

}

**Required annotations:**

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

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="

http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">

<!-- To activate the '@Resource' annotation in the spring framework -->

<context:annotation-config />

<bean id="mycompany" class="com.spring.pojo.Company">

<property name="name" value="Test Pvt. Ltd." />

<property name="location" value="India" />

</bean>

<bean id="myemployee" class="com.spring.pojo.Employee">

<property name="id" value="123456" />

<property name="name" value="Charlotte O' Neil" />

</bean>

</beans>

@insert annotation:

**Runmyprogram.java**

package com.springexample;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class RunMyProgram {

    public static void main(String[] args) {

        ApplicationContext context = new ClassPathXmlApplicationContext("beans.xml");

        StudentHolder studentHolder = (StudentHolder) context.getBean("studentHolder");

            studentHolder.displayStudentDetails();

    }

}

Studentholder.java

package com.springexample;

import javax.inject.Inject;

public class StudentHolder {

    /\* Inject annotation wires the property byType by default \*/

    @Inject

    Student student;

    public Student getStudent() {

        return student;

    }

    public void setStudent(Student student) {

        this.student = student;

    }

    public void displayStudentDetails(){

        System.out.println("Student Details");

        System.out.println("---------------");

        System.out.println("Student No: "+student.getStudentNo());

        System.out.println("Student Name: "+student.getStudentName());

    }

}

Beans.xml

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

<beans xmlns="<http://www.springframework.org/schema/beans>"

    xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>"

    xmlns:context="<http://www.springframework.org/schema/context>"

    xsi:schemaLocation="<http://www.springframework.org/schema/beans> <http://www.springframework.org/schema/beans/spring-beans-3.0.xsd>

<http://www.springframework.org/schema/context> <http://www.springframework.org/schema/context/spring-context-3.0.xsd>">

    <bean id="stu" class="com.springexample.Student">

        <property name="studentNo" value="1001" />

        <property name="studentName" value="John Peter" />

    </bean>

    <bean id="studentHolder" class="com.springexample.StudentHolder" />

    <context:annotation-config />

</beans>

Student.java

package com.springexample;

public class Student {

    private int studentNo;

    private String studentName;

    public int getStudentNo() {

        return studentNo;

    }

    public void setStudentNo(int studentNo) {

        this.studentNo = studentNo;

    }

    public String getStudentName() {

        return studentName;

    }

    public void setStudentName(String studentName) {

        this.studentName = studentName;

    }

}

**7.** **Write a java program to demonstrate SPEL (Spring ExpressionLanguage ).**

**Speldemo.java**

**package** Assignment7\_;

**import** org.springframework.expression.Expression;

**import** org.springframework.expression.ExpressionParser;

**import** org.springframework.expression.spel.standard.SpelExpressionParser;

**public** **class** SPEL\_demo {

**public** **static** **void** main(String[] args) {

ExpressionParser expressionParser = **new** SpelExpressionParser();

//1. Literal Expression

Expression expression = expressionParser.parseExpression(" 'Hello World' ");

String str=expression.getValue(String.**class**);

System.***out***.println(" Literal Expression value : " +str);

// 2. Method Invocation

expression = expressionParser.parseExpression(" 'Hello ' .concat('world')");

str=expression.getValue(String.**class**);

System.***out***.println(" Method concatnation : " +str);

// 3. Mathematical operator

expression = expressionParser.parseExpression(" 10+5");

Integer value = expression.getValue(Integer.**class**);

System.***out***.println("Addition of two number : " +value);

expression = expressionParser.parseExpression(" 10-5");

value = expression.getValue(Integer.**class**);

System.***out***.println("Substraction of two number : " +value);

expression = expressionParser.parseExpression(" 10\*5");

value = expression.getValue(Integer.**class**);

System.***out***.println("Multiplication of two number : " +value);

expression = expressionParser.parseExpression(" 10/5");

value = expression.getValue(Integer.**class**);

System.***out***.println("Division of two number : " +value);

// 4. Relational Operator

expression = expressionParser.parseExpression(" 5 < 8");

**boolean** ans = expression.getValue(Boolean.**class**);

System.***out***.println(" Answer = " +ans);

expression = expressionParser.parseExpression(" 7 > 8");

ans = expression.getValue(Boolean.**class**);

System.***out***.println(" Answer = " +ans);

// 5. Logical operator

expression = expressionParser.parseExpression("900 > 500 && 200 <500 ");

ans = expression.getValue(Boolean.**class**);

System.***out***.println("Logical Operator Answer = " +ans);

// 6. ternary operator

expression = expressionParser.parseExpression(" 'some value ' != null ? 'some value ' : 'default' ");

str = expression.getValue(String.**class**);

System.***out***.println("ternary Operator Answer = " +str);

// 7. Elvis Operator

expression = expressionParser.parseExpression(" 'some value' ?: 'default'");

str = expression.getValue(String.**class**);

System.***out***.println("Elvis Operator Answer = " +str);

// 8. Regex/matches operator

expression = expressionParser.parseExpression(" ' UPPERCASE STRING' matches '[A-Z\\s]+'");

ans = expression.getValue(Boolean.**class**);

System.***out***.println(" matches Answer = " +ans);

expression = expressionParser.parseExpression(" ' lower case' matches '[a-z\\s]+'");

ans = expression.getValue(Boolean.**class**);

System.***out***.println(" matches Answer = " +ans);

}

}

**Sample.java**

**package** Assignment7\_;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**public** **class** Sample

{

**private** String str="PalakGupta";

**private** ArrayList<Integer> list = **new** ArrayList<>();

**private** HashMap<String , String > map = **new** HashMap<>();

**public** Sample()

{

list.add(1);

list.add(6);

list.add(9);

list.add(8);

map.put("one", " value 1");

map.put("two", " value 2");

map.put("three", " value 3");

map.put("four", " value 4");

}

**public** String getStr() {

**return** str;

}

**public** **void** setStr(String str) {

**this**.str = str;

}

**public** ArrayList<Integer> getList() {

**return** list;

}

**public** **void** setList(ArrayList<Integer> list) {

**this**.list = list;

}

**public** HashMap<String, String> getMap() {

**return** map;

}

**public** **void** setMap(HashMap<String, String> map) {

**this**.map = map;

}

}

**SampleTest.java**

**package** Assignment7\_;

**import** org.springframework.expression.Expression;

**import** org.springframework.expression.ExpressionParser;

**import** org.springframework.expression.spel.standard.SpelExpressionParser;

**import** org.springframework.expression.spel.support.StandardEvaluationContext;

**public** **class** Test\_Sample {

**public** **static** **void** main(String[] args)

{

ExpressionParser expressionParser = **new** SpelExpressionParser();

// creating evaluationcontext from bean

Sample sample = **new** Sample();

StandardEvaluationContext testContext = **new** StandardEvaluationContext(sample);

// Str value

Expression expression = expressionParser.parseExpression("str");

String strval=expression.getValue(testContext , String.**class**);

System.***out***.println(" str value : " +strval);

// compare

// Str value

expression = expressionParser.parseExpression("str == 'PalakGupta'");

Boolean boolval=expression.getValue(testContext , Boolean.**class**);

System.***out***.println(" compare str value : " +boolval);

// List :

expression = expressionParser.parseExpression("list[2]");

strval=expression.getValue(testContext , String.**class**);

System.***out***.println(" List : " +strval);

// map

expression = expressionParser.parseExpression("map['two']");

strval=expression.getValue(testContext , String.**class**);

System.***out***.println(" Map : " +strval);

}

}

**8. Write a java program to demonstrate InitializingBean and DisponsableBean.**

**TRY different Ways:**

**(using init-method and Destroy method xml config file )**

**(Use @PostConstruct and @preDestroy)**

**1 Way:**

**ANS:**

**Class 1:**

**package** QuestionEight;

**import** org.springframework.beans.factory.DisposableBean;

**import** org.springframework.beans.factory.InitializingBean;

**public** **class** CustomerService **implements** InitializingBean, DisposableBean {

**private** String msg;

**public** String getMsg() {

**return** msg;

}

**public** **void** setMsg(String msg) {

**this**.msg = msg;

}

**public** **void** destroy() **throws** Exception {

// **TODO** Auto-generated method stub

System.***out***.println("Spring Container is destroy! Customer clean up");

}

**public** **void** afterPropertiesSet() **throws** Exception {

// **TODO** Auto-generated method stub

System.***out***.println("Init method after properties are set : " + msg);

}}

**Class 2:**

**package** QuestionEight;

**import** org.springframework.context.ConfigurableApplicationContext;

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

**public** **class** Snippet {

**public** **static** **void** main(String[] args) {

ConfigurableApplicationContext context = **new** ClassPathXmlApplicationContext(

**new** String[] { "SpringCustomer.xml" });

CustomerService cust = (CustomerService) context.getBean("customerService");

System.***out***.println(cust);

context.close();

}

}

**XML FILE**

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

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans-2.5.xsd"*>

<bean id=*"customerService"* class=*"QuestionEight.CustomerService"*>

<property name=*"msg"* value=*"i'm property message"* />

</bean>

</beans>

OUTPUT:

Init method after properties are set : i'm property message

Spring Container is destroy! Customer clean up

**2nd Way:**

**Class 1:**

**package** EightB;

**import** javax.annotation.PostConstruct;

**import** javax.annotation.PreDestroy;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Component;

@Component

**public** **class** MyBean {

**public** MyBean() {

System.***out***.println("MyBean instance created");

}

@PostConstruct

**private** **void** init() {

System.***out***.println("Verifying Resources");

}

@PreDestroy

**private** **void** shutdown() {

System.***out***.println("Shutdown All Resources");

}

**public** **void** close() {

System.***out***.println("Closing All Resources");

}

}

**Class 2:**

**package** EightB;

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

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

**import** org.springframework.context.annotation.Scope;

@Configuration

**public** **class** MyConfiguration {

@Bean

@Scope(value = "singleton")

**public** MyBean myBean() {

**return** **new** MyBean();

}

}

**Class 3:**

**package** EightB;

**import** org.springframework.context.annotation.AnnotationConfigApplicationContext;

**public** **class** SpringApp {

**public** **static** **void** main(String[] args) {

AnnotationConfigApplicationContext ctx = **new** AnnotationConfigApplicationContext();

ctx.register(MyConfiguration.**class**);

ctx.refresh();

MyBean mb1 = ctx.getBean(MyBean.**class**);

System.***out***.println(mb1.hashCode());

MyBean mb2 = ctx.getBean(MyBean.**class**);

System.***out***.println(mb2.hashCode());

ctx.close();

}

}

**OUTPUT:**

MyBean instance created

Verifying Resources

2145970759

2145970759

Shutdown All Resources

Closing All Resources

**9. Write a Java program to demonstrate complete Bean Life cycle .**

**ANS:**

**public** **class** HelloWorld {

**public** **void** init() **throws** Exception

{

System.***out***.println(

"Bean HelloWorld has been "

+ "instantiated and I'm "

+ "the init() method");

}

// This method executes

// when the spring container

// is closed

**public** **void** destroy() **throws** Exception

{

System.***out***.println(

"Container has been closed "

+ "and I'm the destroy() method");

}

}

Class 2: Main method Class

**import** org.springframework.context.ConfigurableApplicationContext;

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

**public** **class** Client {

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

// Loading the Spring XML configuration

// file into the spring container and

// it will create the instance of

// the bean as it loads into container

ConfigurableApplicationContext cap = **new** ClassPathXmlApplicationContext("spring.xml");

// It will close the spring container

// and as a result invokes the

// destroy() method

cap.close();

}

}

XML file :

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd"*>

<bean id=*"HelloBean"* class=*"springcore\_example.HelloBean"*>

<property name =*"name"* value=*"sarika"*>

</property>

</bean>

</beans>

**OUTPUT:**

Bean HelloWorld has been instantiated and I'm the init() method

Container has been closed and I'm the destroy() method

**10. Write a Java program to demonstrate ApplicationContextAware interface .**

**ANS:**

**Class 1:**

**package** AplicationContextAware;

**public** **class** Employee {

**private** String Name;

**public** String getName() {

**return** Name;

}

**public** **void** setName(String name) {

Name = name;

}

@Override

**public** String toString() {

**return** "employee [Name=" + Name + "]";

}

}

Class 2: here ApplicationContextAware interface is used

**package AplicationContextAware;**

**import org.springframework.beans.BeansException;**

**import org.springframework.context.ApplicationContext;**

**import org.springframework.context.ApplicationContextAware;**

**public class AppContextAwareImpl implements ApplicationContextAware {**

**private ApplicationContext applicationContext;**

**public void setApplicationContext(ApplicationContext applicationContext) throws BeansException {**

**// TODO Auto-generated method stub**

**System.out.println("set Application Context method is called by the spring container");**

**this.applicationContext = applicationContext;**

**}**

**public void displayEmployeeDetails() {**

**Employee employee = applicationContext.getBean("employee", Employee.class);**

**System.out.println("Got employee object from the applicationContext(Spring Container)=" + employee);**

**System.out.println("is employee object Singleton =" + applicationContext.isSingleton("employee"));**

**}**

**}**

**Class3:**

**package** AplicationContextAware;

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

**public** **class** App {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

ClassPathXmlApplicationContext applicationContext= **new** ClassPathXmlApplicationContext("applicationContext.xml");

AppContextAwareImpl applicationContextAwareImpl= applicationContext.getBean("applicationContextAware",AppContextAwareImpl.**class**);

applicationContextAwareImpl.displayEmployeeDetails();

applicationContext.close();

}

}

**XML file: **

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

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans*

*http://www.springframework.org/schema/beans/spring-beans.xsd"*>

<bean id=*"employee"* class=*"AplicationContextAware.Employee"*>

<property name=*"name"* value =*"peter"* />

</bean>

<bean id=*"applicationContextAware"* class=*"AplicationContextAware.AppContextAwareImpl"*></bean>

</beans>

**OUTPUT:**

set Application Context method is called by the spring container

Got employee object from the applicationContext(Spring Container)=employee [Name=peter]

is employee object Singleton =true