Q1. **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**

Ans🡪Customer.java

**package** org.spring.example1;

**public** **class** Customer {

**private** **int** customer\_id;

**private** String customer\_name;

**private** **int** customer\_contact;

**private** Address street;

**private** Address city;

**private** Address state;

**private** Address zip;

**private** Address country;

**public** **int** getCustomer\_id() {

**return** customer\_id;

}

**public** **void** setCustomer\_id(**int** customer\_id) {

**this**.customer\_id = customer\_id;

}

**public** String getCustomer\_name() {

**return** customer\_name;

}

**public** **void** setCustomer\_name(String customer\_name) {

**this**.customer\_name = customer\_name;

}

**public** **int** getCustomer\_contact() {

**return** customer\_contact;

}

**public** **void** setCustomer\_contact(**int** customer\_contact) {

**this**.customer\_contact = customer\_contact;

}

**public** Address getStreet() {

**return** street;

}

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

**this**.street = street;

}

**public** Address getCity() {

**return** city;

}

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

**this**.city = city;

}

**public** Address getState() {

**return** state;

}

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

**this**.state = state;

}

**public** Address getZip() {

**return** zip;

}

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

**this**.zip = zip;

}

**public** Address getCountry() {

**return** country;

}

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

**this**.country = country;

}

**public** **void** display() {

System.***out***.println("CustomerId: " +getCustomer\_id()+ " "+ "CustomerName: "+getCustomer\_name()+ " "+"CustomerContact: "+getCustomer\_contact());

System.***out***.println("CustomerAddress: "+getStreet().getStreet());

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

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

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

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

}

}

Address.java

**package** org.spring.example1;

**public** **class** Address {

**private** String street;

**private** String city;

**private** String state;

**private** **int** zip;

**private** String country;

**public** String getStreet() {

**return** street;

}

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

**this**.street = street;

}

**public** String getCity() {

**return** city;

}

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

**this**.city = city;

}

**public** String getState() {

**return** state;

}

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

**this**.state = state;

}

**public** **int** getZip() {

**return** zip;

}

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

**this**.zip = zip;

}

**public** String getCountry() {

**return** country;

}

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

**this**.country = country;

}

}

Test.java

**package** org.spring.example1;

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

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

**public** **class** Test {

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

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

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

customer.display();

}

}

Customer.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=*"Customer"* class=*"org.spring.example1.Customer"*>

<property name=*"customer\_id"* value=*"1"*/>

<property name=*"customer\_name"* value=*"Chaitu"*/>

<property name=*"customer\_contact"* value=*"85632"*/>

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

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

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

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

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

</bean>

<bean id=*"CustomerAddress"* class=*"org.spring.example1.Address"*>

<property name=*"street"* value=*"25"*/>

<property name=*"city"* value=*"Pune"*/>

<property name=*"state"* value=*"Maharashtra"*/>

<property name=*"zip"* value=*"411012"*/>

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

</bean>

</beans>

OutPut:

log4j:WARN No appenders could be found for logger (org.springframework.core.env.StandardEnvironment).

log4j:WARN Please initialize the log4j system properly.

CustomerId: 1 CustomerName: Chaitu CustomerContact: 85632

CustomerAddress: 25

Pune

Maharashtra

411012

India

Q2. **Example of injecting collections….**

Ans🡪

**package** org.spring.example2;

**import** java.util.List;

**import** java.util.Map;

**import** java.util.Set;

**public** **class** Question {

List questionList;

Set questionSet;

Map questionMap;

**public** List getQuestionList() {

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

**return** questionList;

}

**public** **void** setQuestionList(List questionList) {

**this**.questionList = questionList;

}

**public** Set getQuestionSet() {

System.***out***.println("Set: "+ questionSet);

**return** questionSet;

}

**public** **void** setQuestionSet(Set questionSet) {

**this**.questionSet = questionSet;

}

**public** Map getQuestionMap() {

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

**return** questionMap;

}

**public** **void** setQuestionMap(Map questionMap) {

**this**.questionMap = questionMap;

}

}

**package** org.spring.example2;

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

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

**public** **class** Test {

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

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

Question question= (Question) context.getBean("Question");

question.getQuestionList();

question.getQuestionSet();

question.getQuestionMap();

}

}

<?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=*"Question"* class=*"org.spring.example2.Question"*>

<property name=*"questionList"*>

<list>

<value>1</value>

<value>What is your name</value>

<value>Chaitu</value>

</list>

</property>

<property name=*"questionSet"*>

<set>

<value>1</value>

<value>What is your name</value>

<value>Chaitu</value>

</set>

</property>

<property name=*"questionMap"*>

<map>

<entry key=*"1"* value=*"1"*/>

<entry key=*"2"* value=*"what is your name"*/>

<entry key=*"3"* value=*"chaitu"*/>

</map>

</property>

</bean>

</beans>

Output:

log4j:WARN No appenders could be found for logger (org.springframework.core.env.StandardEnvironment).

log4j:WARN Please initialize the log4j system properly.

List: [1, What is your name, Chaitu]

Set: [1, What is your name, Chaitu]

Map: {1=1, 2=what is your name, 3=chaitu}

Q3.

Ans🡪

Q4. **Example on @controller @service @repository @autowired @configuration @bean**

**Ans🡪**

@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() );

}

}

@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 2:10:56 PM org.springframework.context.support.AbstractApplicationContext prepareRefresh

INFO: Refreshing org.springframework.context.support.ClassPathXmlApplicationContext@31610302: startup date [Tue Jan 25 14:10:56 IST 2022]; root of context hierarchy

Jan 25, 2022 2:10:56 PM org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions

INFO: Loading XML bean definitions from class path resource [spring.xml]

draw circle

**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;

    }

}

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

**Ans🡪**

**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);

}

}

**Q 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

Q9. Write a java program to demonstrate complete bean life cycle.

Ans🡪

**package** org.spring.example9;

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

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

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

**private** Point pointA;

**private** Point pointB;

**private** Point pointC;

**public** Point getPointA() {

**return** pointA;

}

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

**this**.pointA = pointA;

}

**public** Point getPointB() {

**return** pointB;

}

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

**this**.pointB = pointB;

}

**public** Point getPointC() {

**return** pointC;

}

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

**this**.pointC = pointC;

}

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

System.***out***.println("Point A = ("+getPointA().getX()+","+getPointA().getY()+")");

System.***out***.println("Point B = ("+getPointB().getX()+","+getPointB().getY()+")");

System.***out***.println("Point B = ("+getPointC().getX()+","+getPointC().getY()+")");

}

@Override

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

System.***out***.println("InitializingBean init method called for Triangle");

}

@Override

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

System.***out***.println("DisposibleBean destroy method for the Triangle");

}

}

**package** org.spring.example9;

**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;

}

}

<?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=*"triangle"* class=*"org.spring.example9.Triangle"* autowire=*"byName"*>

</bean>

<bean id=*"pointA"* class=*"org.spring.example9.Point"*>

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

<property name=*"y"* value=*"0"*></property>

</bean>

<bean id=*"pointB"* class=*"org.spring.example9.Point"*>

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

<property name=*"y"* value=*"0"*></property>

</bean>

<bean id=*"pointC"* class=*"org.spring.example9.Point"*>

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

<property name=*"y"* value=*"0"*></property>

</bean>

</beans>

**package** org.spring.example9;

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

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

**public** **class** DrawingApp {

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

AbstractApplicationContext context= **new** ClassPathXmlApplicationContext("spring2.xml");

context.registerShutdownHook();

Triangle triangle = (Triangle) context.getBean("triangle");

triangle.draw();

}

}

Output:

log4j:WARN No appenders could be found for logger (org.springframework.core.env.StandardEnvironment).

log4j:WARN Please initialize the log4j system properly.

InitializingBean init method called for Triangle

Point A = (0,0)

Point B = (-20,0)

Point B = (20,0)

DisposibleBean destroy method for the Triangle

Q.10 Write a java program to demonstrate AppllicationContextAware interface

Ans🡪

**package** org.spring.example10;

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

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

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

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

**public** **class** Triangle **implements** ApplicationContextAware ,BeanNameAware{

**private** Point pointA;

**private** Point pointB;

**private** Point pointC;

**private** ApplicationContext context = **null**;

**public** Point getPointA() {

**return** pointA;

}

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

**this**.pointA = pointA;

}

**public** Point getPointB() {

**return** pointB;

}

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

**this**.pointB = pointB;

}

**public** Point getPointC() {

**return** pointC;

}

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

**this**.pointC = pointC;

}

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

System.***out***.println("Point A = ("+getPointA().getX()+","+getPointA().getY()+")");

System.***out***.println("Point B = ("+getPointB().getX()+","+getPointB().getY()+")");

System.***out***.println("Point B = ("+getPointC().getX()+","+getPointC().getY()+")");

}

@Override

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

**this**.context=context;

}

@Override

**public** **void** setBeanName(String beanName) {

System.***out***.println("Bean name is:"+ beanName);

}

}

**package** org.spring.example10;

**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;

}

}

Spring2.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=*"triangle"* class=*"org.spring.example10.Triangle"* autowire=*"byName"*>

</bean>

<bean id=*"pointA"* class=*"org.spring.example10.Point"*>

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

<property name=*"y"* value=*"0"*></property>

</bean>

<bean id=*"pointB"* class=*"org.spring.example10.Point"*>

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

<property name=*"y"* value=*"0"*></property>

</bean>

<bean id=*"pointC"* class=*"org.spring.example10.Point"*>

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

<property name=*"y"* value=*"0"*></property>

</bean>

</beans>

**package** org.spring.example10;

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

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

**public** **class** DrawingApp {

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

ApplicationContext context= **new** ClassPathXmlApplicationContext("spring2.xml");

Triangle triangle = (Triangle) context.getBean("triangle");

triangle.draw();

}

}

Output: log4j:WARN No appenders could be found for logger (org.springframework.core.env.StandardEnvironment).

log4j:WARN Please initialize the log4j system properly.

Bean name is:triangle

Point A = (0,0)

Point B = (-20,0)

Point B = (20,0)