1. Create a address class with following attribute.

Class Address

package spring\_example;

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;

}

}

Class Customer

package spring\_example;

import java.util.List;

public class Customer {

private int customerid;

private String customername;

private String customercontact;

private List<Address> address;

public List<Address> getAddress() {

return address;

}

public void setAddress(List<Address> address) {

this.address = address;

}

public int getCustomerid() {

return customerid;

}

public void setCustomerid(int customerid) {

this.customerid = customerid;

}

public String getCustomername() {

return customername;

}

public void setCustomername(String customername) {

this.customername = customername;

}

public String getCustomercontact() {

return customercontact;

}

public void setCustomercontact(String customercontact) {

this.customercontact = customercontact;

}

public void cust()

{

System.out.println();

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*customer Details\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println();

System.out.println("Id: "+this.customerid);

System.out.println("Name: "+this.customername);

System.out.println("Contact no: "+this.customercontact);

for(Address ad : address)

{

System.out.println("Address : Street:"+ad.getStreet()+", City:"+ad.getCity()+", State:"+ad.getState()+", Zipcode:"+ad.getZip()+", Country:"+ad.getCountry());

}

}

}

Beans.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="Customer" class="spring\_example.Customer" >

<property name="customerid" value="632"/>

<property name="customername" value="AJAY PRADHAN"/>

<property name="customercontact" value="9096660872"/>

<property name="address">

<list>

<ref bean="AddressDetails"/>

</list>

</property>

</bean>

<bean id="AddressDetails" class="spring\_example.Address" >

<property name="street" value="Hingna Road"/>

<property name="city" value="Nagpur"/>

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

<property name="zip" value="440016"/>

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

</bean>

</beans>

Main Test CustTest Class

package spring\_example;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class CustTest {

@SuppressWarnings("unused")

private static ApplicationContext context;

public static void main(String a[])

{

@SuppressWarnings("resource")

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

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

ct.cust();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2.Example of injecting collection.

##### 01 Class:Question

import java.util.Iterator;

import java.util.List;

import java.util.Map;

import java.util.Set;

public class Question {

private int QuestionID;

private String Question;

private List<String> Answer;

private Set<String> Answer1;

private Map<Integer,String> Answer2;

public Question(int questionID, String question, List<String> answer, Set<String> answer1, Map<Integer,String> answer2) {

super();

QuestionID = questionID;

Question = question;

Answer = answer;

Answer1=answer1;

Answer2=answer2;

}

public void showAnswer() {

System.out.println(QuestionID+ " "+ Question);

System.out.println("Answers from(List,Set, Map)");

System.out.println("Answers from(List)");

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

while(i.hasNext()) {

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

}

System.out.println("Answers from(Set)");

Iterator<String> iSet=Answer1.iterator();

while(iSet.hasNext()) {

System.out.println(iSet.next());

}

System.out.println("Answers from(Map)");

for(Map.Entry<Integer, String> entry: Answer2.entrySet()) {

System.out.println(entry.getKey()+ " "+ entry.getValue());

}

}

}

###Class:02 TestQuestion

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class TestQuestion {

private static ApplicationContext context;

public static void main(String[] args) {

// TODO Auto-generated method stub

context=new ClassPathXmlApplicationContext("QuestionXML.xml");

Question q=(Question) context.getBean("que");

q.showAnswer();

}

}

#### 03 Class:QuestionXML

<?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="que" class="Question">

<constructor-arg value="20"></constructor-arg>

<constructor-arg value="Why Spring is Popular..?" />

<constructor-arg>

<list>

<value>1. It Support All Framework</value>

<value>2. Spring is used for Enterprise Level Application</value>

</list>

</constructor-arg>

<constructor-arg>

<set>

<value>1. All the frameworks are supported by spring like Structs,Hibernate and Jsf.</value>

<value>2. simple learning curves</value>

</set>

</constructor-arg>

<constructor-arg>

<map>

<entry key="1" value="Spring Framework" />

<entry key="2" value="Spring MVC Framework" />

</map>

</constructor-arg>

</bean>

</beans>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3. Example on autowiring.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4.Example on @Controller, @Service, @Repository, @Autowired, @Configuration and @Bean

Modify the above application, use annotations and java based configuration.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5. Write a program to demostrate use of @Resource, @Inject, @Required annotations.

class- Employee.java

package pojo;

import javax.annotation.Resource;

public class Employee {

private String id;

private String name;

private String address;

@Required

public void setName(String name) {

this.name = name;

}

public String getName() {

return name;

}

@Inject

private Address address;

@Resource(name="mycompany")

private Company company;

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public Company getCompany() {

return company;

}

public void setCompany(Company company) {

this.company = company;

}

public Address getAddress() {

return address;

}

public void setAddress(Address address) {

this.address = address;

}

@Override

public String toString() {

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

}

}

class- AppMain.java

package pojo;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import 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());

}

}

class- resource-annotation.xml

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

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

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

xsi:schemaLocation="

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

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

<!-- 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>

<bean

class="org.springframework.beans.factory.

annotation.AutowiredAnnotationBeanPostProcessor">

</bean>

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

</bean>

</beans>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

6. Example of @Component, @Value, @PropertySource ,@Environment.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

7.write a java program to demonstrate SPEL(Spring Expression language).

package spring\_core;

import org.springframework.expression.Expression;

import org.springframework.expression.ExpressionParser;

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

public class example\_two {

public static void main(String[] args) {

// TODO Auto-generated method stub

ExpressionParser parser = new SpelExpressionParser();

Expression exp = parser.parseExpression("'Barnali'");

String message = (String) exp.getValue();

System.out.println(message);

//OR

//System.out.println(parser.parseExpression("'Hello SPEL'").getValue());

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

8. Write a java program to demonstrate InitializingBean and DisposableBean.

a)(Use init method and destroy method in xml config)

Employee.java

import org.springframework.beans.factory.DisposableBean;

import org.springframework.beans.factory.InitializingBean;

public class Employee implements InitializingBean,DisposableBean{

private String name;

private int age;

public Employee() {

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public void afterPropertiesSet() throws Exception{

System.out.println("Initializing ");

}

public void destroy() throws Exception{

System.out.println("Destroying Employee Bean");

}

public void display() {

System.out.println("Name of Employee is : "+this.getName());

System.out.println("Age is : "+this.getAge());

}

}

Main.java

import org.springframework.context.support.AbstractApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main {

public static void main(String[] args) {

AbstractApplicationContext context=new ClassPathXmlApplicationContext(("spring.xml"));

context.registerShutdownHook();

Employee emp=(Employee) context.getBean("employee");

emp.display();

}

}

spring.xml

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

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

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

xsi:schemaLocation="

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

<bean id="employee" class="spring\_example.Employee">

<property name="name" value="David"/>

<property name="age" value="27"/>

</bean>

</beans>

b) Use @PostConstruct and @Preconstruct

import javax.annotation.PostConstruct;

import javax.annotation.PreDestroy;

public class Employee2 {

private String name;

private int age;

public Employee2() {

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

@PostConstruct

public void initializing() {

System.out.println("Initializing Employee Been");

}

@PreDestroy

public void destroying() {

System.out.println("Destroying Employee Bean");

}

}

Main

import org.springframework.context.support.AbstractApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Main2 {

public static void main(String[] args) {

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

context.registerShutdownHook();

Employee2 emp=(Employee2) context.getBean("employee");

System.out.println("Employee name: "+emp.getName());

System.out.println("Age is: "+emp.getAge());

}

}

spring

<?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"

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

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="employee" class="spring\_example.Employee2">

<property name="name" value="David"/>

<property name="age" value="27"/>

</bean>

</beans>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

9. Write a java program to demonstrate Complete Bean Life cycle.

Triangle.java

package com.bean2;

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 void setPointA(Point pointA) {

this.pointA = pointA;

}

public void setPointB(Point pointB) {

this.pointB = pointB;

}

public void setPointC(Point pointC) {

this.pointC = pointC;

}

public void draw()

{

System.out.println("PointA is ("+pointA.getX()+", "+pointA.getY()+")");

System.out.println("PointB is ("+pointB.getX()+", "+pointB.getY()+")");

System.out.println("PointC is ("+pointC.getX()+", "+pointC.getY()+")");

}

@Override

public void afterPropertiesSet() throws Exception

{

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

}

@Override

public void destroy() throws Exception

{

System.out.println("DisposableBean destroy method is called for Triangle");

}

public void myInit()

{

System.out.println("My init method is called for Triangle");

}

public void cleanUp()

{

System.out.println("cleanUp method is called for Triangle");

}

}

Point.java:

package com.bean2;

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;

}

}

}

DrawingApp.java:

package com.bean2;

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("spring.xml");

context.registerShutdownHook();

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

triangle.draw();

}

}

}

Spring.xml:

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

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

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

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

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

<bean autowire="byName" class="com.bean2.Triangle" destroy-method="cleanUp" id="triangle" init-method="myInit">

</bean>

<bean class="com.bean2.Point" id="pointA">

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

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

</bean>

<bean class="com.bean2.Point" id="pointB">

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

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

</bean>

<bean class="com.bean2.Point" id="pointC">

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

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

</bean>

</beans>

Output:

InitialzingBean init method is called for Triangle

My init method is called for Triangle

PointA is(0,0)

PointB is(-20,0)

PointC is (20,0)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

10.Write a java program to demonstrate ApplicationContextAware interface.

Triangle.java

package spring.pakage;

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;

public void setPointA(Point pointA) {

this.pointA = pointA;

}

public void setPointB(Point pointB) {

this.pointB = pointB;

}

public void setPointC(Point pointC) {

this.pointC = pointC;

}

public void draw()

{

System.out.println("PointA is ("+pointA.getX()+", "+pointA.getY()+")");

System.out.println("PointB is ("+pointB.getX()+", "+pointB.getY()+")");

System.out.println("PointC is ("+pointC.getX()+", "+pointC.getY()+")");

}

@Override

public void setApplicationContext(ApplicationContext context) throws BeansException

{

}

@Override

public void setBeanName(String beanName)

{

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

}

}

Point.java

package spring.pakage;

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;

}

}

spring.xml

<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="triangle" class="spring.pakage.Triangle"></bean>

<bean class="spring.pakage.Triangle" id="pointA">

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

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

</bean>

<bean class="spring.pakage.Triangle" id="pointB">

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

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

</bean>

<bean class="spring.pakage.Triangle" id="pointC">

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

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

</bean>

</beans>

DrawingApp.java

package spring.pakage;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class DrawingApp

{

private static ApplicationContext context;

public static void main(String[] args)

{

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

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

triangle.draw();

}

}

output:-

Bean name is: triangle

PointA is (0, 0)

PointB is (-20, 0)

PointC is (20, 0)