**\*\*\*\*\*\*\*VaishaliSpring2.5XmlProject\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*IGreet.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

public interface IGreet

{

public String greetMe();

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*BirthDayWishImpl.java\*\*\*\*\*\*\*\*\*\*\*\***

**package** com.cg.bean;

**public** **class** BirdayWishImpl **implements** IGreet

{

@Override

**public** String greetMe()

{ **return** " Happy Birth day "; }

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*NewYearWishImpl.java\*\*\*\*\*\*\*\*\***

package com.cg.bean;

import org.springframework.beans.BeansException;

import org.springframework.beans.factory.BeanFactory;

import org.springframework.beans.factory.BeanFactoryAware;

import org.springframework.beans.factory.BeanNameAware;

import org.springframework.beans.factory.DisposableBean;

import org.springframework.beans.factory.InitializingBean;

public class NewYearWishGreetImpl

implements IGreet,BeanNameAware ,BeanFactoryAware ,DisposableBean ,InitializingBean

{

String name;

String year;

public String getYear() { return year; }

public void setYear(String year) {

this.year = year;

System.out.println("\*\*\*\*\*\*\*\*setYear() called and year initialized\*\*\*");

}

public String getName() { return name; }

public void setName(String name) {

this.name = name;

System.out.println("\*\*\*\*\*\*\*\*setName() called and name initialized\*\*\*");

}

public NewYearWishGreetImpl()

{

System.out.println("\*\*\*\*NewYearWishGreetImpl Bean Instantiated\*\*\* ");

}

@Override

public String greetMe()

{

return name +" Happy New Year "+year;

}

@Override

public void setBeanFactory(BeanFactory bf)

throws BeansException {

System.out.println("setBeanFactory callled \*\*\*\*\*");

}

@Override

public void setBeanName(String bn) {

System.out.println("setBeanName callled \*\*\*\*\*"+bn);

}

public void initGreetAfterCreation()

{

System.out.println(" \*\*After bean instantiated "+

"some initialization can be performed to get it to a usable state "+

" that time initGreetAfterCreation is called\*\*\*\*\*\*\*\*\*\*\*\*");

}

public void destroyGreetBeforeDeletion()

{

System.out.println(" \*\*When the bean is removed from"+

" the container, some cleanup may be required"+

" that time destroyGreetBeforeDeletion is called\*\*\*\*\*\*\*\*\*\*\*\*");

}

@Override

public void destroy() throws Exception {

System.out.println("bean is destroied from container\*\*\*\*\*\*");

}

@Override

public void afterPropertiesSet() throws Exception {

System.out.println(" This Is called After All PropertiesSet ");

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*NewYearWishPostProcessor.java\*\*\*\***

package com.cg.bean;

import org.springframework.beans.BeansException;

import org.springframework.beans.factory.config.BeanPostProcessor;

public class NewYearWishPostProcessor implements BeanPostProcessor {

@Override

public Object postProcessAfterInitialization(Object bean,

String beanName)

throws BeansException {

System.out.println("AfterInitialization : " + beanName);

Class cls=bean.getClass();

String type=cls.getName();

System.out.println(type);

return bean; // you can return any other object as well

}

@Override

public Object postProcessBeforeInitialization(Object bean, String beanName)

throws BeansException {

System.out.println("BeforeInitialization : " + beanName);

return bean; // you can return any other object as well

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ICurrencyConvertor.java\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

public interface ICurrencyConvertor

{

public double dollerToRupees(double dollers);

public int yenToRupees(int yen);

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CurrencyConvertorImpl.java\*\*\*\*\*\***

package com.cg.bean;

public class CurrencyConvertorImpl implements

ICurrencyConvertor

{

private int exchangeRate;

private String countryName;

@Override

public String toString() {

return "CurrencyConvertorImpl [exchangeRate=" + exchangeRate

+ ", countryName=" + countryName + "]";

}

public int getExchangeRate() { return exchangeRate; }

public void setExchangeRate(int exchangeRate)

{

this.exchangeRate = exchangeRate;

}

@Override

public double dollerToRupees(double dollers)

{

return (exchangeRate\*dollers);

}

public String getCountryName() { return countryName; }

public void setCountryName(String countryName) {

this.countryName = countryName;

}

@Override

public int yenToRupees(int yen)

{

return (exchangeRate\*yen);

}

public CurrencyConvertorImpl(int exchangeRate)

{

super();

this.exchangeRate = exchangeRate;

}

public CurrencyConvertorImpl() { super(); }

public CurrencyConvertorImpl(int exchangeRate,

String countryName) {

super();

this.exchangeRate = exchangeRate;

this.countryName = countryName;

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*cg.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:aop=*"http://www.springframework.org/schema/aop"*

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

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

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

*http://www.springframework.org/schema/beans/spring-beans-4.0.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context-3.0.xsd*

*http://www.springframework.org/schema/util*

*http://www.springframework.org/schema/util/spring-util-3.0.xsd*

*"*>

<bean id=*"customEditorConfigurer"*

class=

*"org.springframework.beans.factory.config.CustomEditorConfigurer"*>

<property name=*"customEditors"*>

<map>

<entry key=*"java.sql.Date"*

value=*"com.cg.bean.SQLDateEditor"* />

</map>

</property>

</bean>

<bean

class=*"org.springframework.beans.factory.config.CustomEditorConfigurer"*>

<property name=*"propertyEditorRegistrars"*>

<list>

<bean class=*"com.cg.bean.CustomDateEditorRegistrar"* />

</list>

</property>

</bean>

<util:list id=*"cities"* >

<bean class=*"com.cg.bean.City"* p:name=*"Chicago"* p:population=*"100000"*/>

<bean class=*"com.cg.bean.City"* p:name=*"Atlanta"* p:population=*"200002"*/>

<bean class=*"com.cg.bean.City"* p:name=*"Dallas"* p:population=*"300000"*/>

</util:list>

<!-- \*\*\*\*\*\*\*\*\*\*\*Property File Configuration\*\*\*\*\*\*\*\*\*\* -->

<!-- <util:properties id="userprops" location="classpath:user.properties" /> -->

<!-- \*\*\*\*\*\*\*\*\*\*\*Property File Configuration\*\*\*\*\*\*\*\*\*\* -->

<bean id=*"placeHolderConfig"*

class=*"org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"*>

<property name=*"location"* value=*"user.properties"* />

</bean>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Bean Configuration\*\*\*\*\*\*\* -->

<bean id=*"city1"* class=*"com.cg.bean.City"*>

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

<property name=*"population"* value=*"3000000"* />

</bean>

<bean id=*"city2"* class=*"com.cg.bean.City"*>

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

<property name=*"population"* value=*"5000000"* />

</bean>

<bean id=*"city3"* class=*"com.cg.bean.City"*>

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

<property name=*"population"* value=*"2000000"* />

</bean>

<bean id=*"offAdd"* class=*"com.cg.bean.Address"*>

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

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

<property name=*"zipcode"* value=*"67877"* />

<!-- <property name="city" value="#{city1}"/> -->

<property name=*"city"* ref=*"city1"* />

</bean>

<bean id=*"homeAdd"* class=*"com.cg.bean.Address"*>

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

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

<property name=*"zipcode"* value=*"67877"* />

<property name=*"city"* value=*"#{city2}"* />

</bean>

<!-- Configure Person bean -->

<bean id=*"per"* class=*"com.cg.bean.Person"*>

<property name=*"firstName"* value=*"Vaishali"* />

<property name=*"lastName"* value=*"Srivastava"* />

<property name=*"age"* value=*"80"* />

<property name=*"addList"*>

<list>

<!-- <value>#{homeAdd}</value>

<value>#{offAdd}</value> -->

<ref bean=*"homeAdd"*/>

<ref bean=*"offAdd"*/>

</list>

</property>

</bean>

<bean id=*"obj1"* class=*"com.cg.bean.NewYearWishGreetImpl"* scope=*"prototype"*>

</bean>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<bean id=*"obj2"* class=*"com.cg.bean.BirdayWishImpl"*>

</bean>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*Setter Injection\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<bean id=*"currencyConObj1"* class=*"com.cg.bean.CurrencyConvertorImpl"*>

<property name=*"exchangeRate"* value=*"68"* />

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

</bean>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*Constructor Injection\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<bean id=*"currencyConObj2"* class=*"com.cg.bean.CurrencyConvertorImpl"*>

<constructor-arg index=*"0"* value=*"30"* />

<constructor-arg index=*"1"* value=*"Japan"* />

</bean>

<bean id=*"e1"* class=*"com.cg.bean.Employee"*>

<property name=*"eid"* value=*"#{T(java.lang.Math).random()\*100}"* />

<property name=*"enm"* value=*"#{per.firstName}"* />

<property name=*"eage"* value=*"#{per.getAge()}"* />

<property name=*"empAddress"* ref=*"offAdd"* />

<property name=*"esl"* value=*"#{2345.90}"* />

<property name=*"doj"*><value>getDoj()</value></property>

<property name=*"dob"* value=*"30-09-1999"* />

</bean>

<bean id=*"cList"* class=*"com.cg.bean.CityList"*>

<property name=*"cityList"*>

<list>

<value>#{city1}</value>

<value>#{city2}</value>

<value>#{city3}</value>

</list>

</property>

</bean>

<!-- \*\*\*\*\*\*\*\*\*\*\*\*\*User Bean\*\*\*\*\*\*\*\*\*\*\*\*\*\* -->

<bean id=*"ur"* class=*"com.cg.bean.User"*>

<!-- using EL syntax -->

<!-- <property name="userName" value="#{userprops.username}"/>

<property name="password" value="#{userprops.password}"/>

<property name="userLocs" value="#{cities.![name]}"/> -->

<!-- using Normal syntax -->

<property name=*"userName"* value=*"${username}"*/>

<property name=*"password"* value=*"${password}"*/>

<property name=*"userLocs"* value=*"#{cities.![name]}"*/>

</bean>

</beans>

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* TestSpringCurrClient.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.ui;

import org.springframework.beans.factory.BeanFactory;

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

import org.springframework.core.io.ClassPathResource;

import org.springframework.core.io.Resource;

import com.cg.bean.ICurrencyConvertor;

import com.cg.bean.IGreet;

import com.cg.bean.NewYearWishGreetImpl;

public class TestSpringCurrClient

{

public static void main(String[] args)

{

Resource res=new ClassPathResource("cg.xml");

BeanFactory factory=new XmlBeanFactory(res);

ICurrencyConvertor currC1= (ICurrencyConvertor)factory.

getBean("currencyConObj1");

System.out.println(" 50 Doller Is ? "+

currC1.dollerToRupees(50) + " RS. ");

ICurrencyConvertor currC2= (ICurrencyConvertor)factory.

getBean("currencyConObj2");

System.out.println(" 30 Yen Is ? "+

currC2.yenToRupees(30) + " RS. ");

}}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Address.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

public class Address

{

private String state;

private String country;

private String zipcode;

private City city;

@Override

public String toString() {

return "Address [state=" + state + ", country=" + country

+ ", zipcode=" + zipcode + ", myCity=" + city + "]";

}

public String getState() { return state; }

public void setState(String state) { this.state = state; }

public String getCountry() { return country; }

public void setCountry(String country) { this.country = country; }

public String getZipcode() { return zipcode; }

public void setZipcode(String zipcode) { this.zipcode = zipcode; }

public City getCity() { return city; }

public void setCity(City city) { this.city = city; }

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CityList.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

import java.util.ArrayList;

public class CityList

{

private ArrayList<City> cityList;

@Override

public String toString() {

return "CityList [cityList=" + cityList + "]";

}

public ArrayList<City> getCityList() { return cityList; }

public void setCityList(ArrayList<City> cityList) { this.cityList = cityList; }

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*City.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

public class City

{

private String name;

private int population;

public String getName() { return name; }

public void setName(String name) { this.name = name; }

public int getPopulation() { return population; }

public void setPopulation(int population) { this.population = population; }

@Override

public String toString() {

return "City [name=" + name + ", population=" + population + "]"; }}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Employee.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

import java.sql.Date;

public class Employee

{

private int eid;

private String enm;

private int eage;

private Address empAddress;

Date doj;

java.util.Date dob;

public Date getDoj() { return doj; }

public void setDoj(Date doj) { this.doj = doj; }

public java.util.Date getDob() { return dob; }

public void setDob(java.util.Date dob) { this.dob = dob; }

public Address getEmpAddress() { return empAddress; }

public void setEmpAddress(Address empAddress) { this.empAddress = empAddress; }

private double esl;

@Override

public String toString() {

return "Employee [eid=" + eid + ", enm=" + enm + ", eage=" + eage

+ ", empAddress=" + empAddress + ", doj=" + doj + ", dob="

+ dob + ", esl=" + esl + "]";

}

public int getEid() { return eid; }

public void setEid(int eid) { this.eid = eid; }

public String getEnm() { return enm; }

public void setEnm(String enm) { this.enm = enm; }

public int getEage() { return eage; }

public void setEage(int eage) { this.eage = eage; }

public double getEsl() { return esl; }

public void setEsl(double esl) { this.esl = esl; }

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Person.java\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

import java.util.ArrayList;

public class Person

{

private String firstName;

private String lastName;

private int age;

private ArrayList addList;

public int getAge() { return age; }

public ArrayList getAddList() { return addList; }

public void setAddList(ArrayList addList) { this.addList = addList; }

@Override

public String toString() { return "Person [firstName=" + firstName + ", lastName=" + lastName

+ ", age=" + age + ", addList=" + addList + "]";

}

public void setAge(int age) { this.age = age; }

public String getFirstName() { return firstName; }

public void setFirstName(String firstName) { this.firstName = firstName; }

public String getLastName() { return lastName; }

public void setLastName(String lastName) { this.lastName = lastName; }

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SQLDateEditor.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

import java.beans.PropertyEditorSupport;

import java.sql.Date;

public class SQLDateEditor extends PropertyEditorSupport

{

public SQLDateEditor() { }

public void setAsText(String text) {

if(text.equalsIgnoreCase("getDoj()"))

{

Date date = new Date(System.currentTimeMillis());

setValue(date);

}

else

{

setValue(Date.valueOf(text));

}

}

public String getAsText() {

return getValue().toString(); }}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CustomDateEditorRegistrar.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

import java.text.SimpleDateFormat;

import java.util.Date;

import org.springframework.beans.PropertyEditorRegistrar;

import org.springframework.beans.PropertyEditorRegistry;

import org.springframework.beans.propertyeditors.CustomDateEditor;

public class CustomDateEditorRegistrar implements

PropertyEditorRegistrar

{

@Override

public void registerCustomEditors(

PropertyEditorRegistry registry)

{

registry.registerCustomEditor(

Date.class,

new CustomDateEditor(

new SimpleDateFormat("dd-MM-yyyy"),

false));

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TestSpellClientDemo.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.ui;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.cg.bean.Employee;

import com.cg.bean.Person;

import com.cg.bean.User;

public class TestSpelClientDemo {

public static void main(String[] args)

{

ApplicationContext ctx=

new ClassPathXmlApplicationContext("cg.xml");

Employee tempEmp=(Employee)ctx.getBean("e1");

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*Employee\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println(tempEmp);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*Person\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Person tempPer=(Person)ctx.getBean("per");

System.out.println(tempPer);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*User\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

User tempUser=(User)ctx.getBean("ur");

System.out.println(tempUser);

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*basic.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:aop=*"http://www.springframework.org/schema/aop"*

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

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

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

*http://www.springframework.org/schema/beans/spring-beans-4.0.xsd*

*http://www.springframework.org/schema/context*

*http://www.springframework.org/schema/context/spring-context-3.0.xsd*

*http://www.springframework.org/schema/util*

*http://www.springframework.org/schema/util/spring-util-3.0.xsd*

*"*>

<bean id=*"obj1"* class=*"com.cg.bean.NewYearWishGreetImpl"*

init-method=*"initGreetAfterCreation"*

destroy-method=*"destroyGreetBeforeDeletion"*>

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

<property name=*"year"* value=*"2017"* />

</bean>

<bean class=*"com.cg.bean.NewYearWishPostProcessor"* />

</beans>

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TestSpringLifeCycleClient.java\*\*\*\*\*\*\*\*\*\*\***

package com.cg.ui;

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;

import com.cg.bean.ICurrencyConvertor;

import com.cg.bean.IGreet;

import com.cg.bean.NewYearWishGreetImpl;

public class TestSpringLifeCycleClient

{

public static void main(String[] args)

{

//Resource res=new ClassPathResource("basic.xml");

//BeanFactory factory=new XmlBeanFactory

// (res);

ApplicationContext ctx=

new ClassPathXmlApplicationContext("basic.xml");

IGreet greet1=(IGreet)ctx.getBean("obj1");

System.out.println(greet1.greetMe());

IGreet greet2=(IGreet)ctx.getBean("obj1");

System.out.println(greet2.greetMe());

System.out.println(greet1.hashCode());

System.out.println(greet2.hashCode());

try {

((NewYearWishGreetImpl)greet1).destroy();

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

} }}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*User.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.bean;

import java.util.ArrayList;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.context.annotation.PropertySource;

import org.springframework.context.annotation.PropertySources;

import org.springframework.stereotype.Component;

public class User

{

@Override

public String toString() {

return "User [userName=" + userName + ", password=" + password

+ ", userLocs=" + userLocs + "]";

}

private String userName;

private String password;

private ArrayList<String> userLocs;

public ArrayList getUserLocs() { return userLocs; }

public void setUserLocs(ArrayList userLocs) { this.userLocs = userLocs; }

public String getUserName() { return userName; }

public void setUserName(String userName) { this.userName = userName; }

public String getPassword() { return password; }

public void setPassword(String password) { this.password = password; }

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*TestUserClient.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.ui;

import org.springframework.context.ApplicationContext;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.cg.bean.User;

public class TestUserClient {

public static void main(String[] args)

{

ApplicationContext ctx=new

ClassPathXmlApplicationContext("cg.xml");

User u=(User)ctx.getBean("ur");

System.out.println(" My User Id Id : "+

u.getUserName());

System.out.println(" My User Password Is : "+

u.getPassword());

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*AudienceAspect.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.advices;

import java.util.Arrays;

import org.aspectj.lang.JoinPoint;

import org.aspectj.lang.ProceedingJoinPoint;

import com.cg.aop.bean.Audience;

public class AudienceAspect

{

private Audience audience;

public void setAudience(Audience audience)

{

this.audience = audience;

}

public AudienceAspect() {}

public void performBefore()

{

audience.takeSeates();

audience.turnOffCellPhone();

}

public void performAround(ProceedingJoinPoint joinPoint) throws Throwable {

System.out.println("performAround() is running!");

System.out.println("hijacked method : " + joinPoint.getSignature().getName());

System.out.println("hijacked arguments : " + Arrays.toString(joinPoint.getArgs()));

System.out.println("performAround before is running!");

joinPoint.proceed(); //continue on the intercepted method

System.out.println("performAround after is running!");

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

}

public void performAfterThrowing() {

System.out.println("performAfterThrowing() is running!");

System.out.println("Bool We want our moany back");

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

}

public void performAfter()

{

audience.appLaud();

}}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Audience.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.aop.bean;

public class Audience

{

public void takeSeates()

{

System.*out*.println("Audience is taking their seates.");

}

public void turnOffCellPhone()

{

System.*out*.println("Audience is turnning of their cell phones.");

}

public void appLaud()

{

System.*out*.println("Clap Clap Clap Clap");

}

public void demandRefund()

{

System.*out*.println("Bool We want our moany back");

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*Instrument.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.aop.bean;

public interface Instrument

{

public void play();

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Instrumentalist.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.aop.bean;

public class Instrumentalist implements Performer

{

private String song;

private Instrument instrument;

public Instrumentalist(String song, Instrument instrument) {

super();

this.song = song;

this.instrument = instrument;

}

public Instrumentalist() { super(); }

public String getSong() { return song; }

public void setSong(String song) { this.song = song; }

public Instrument getInstrument() { return instrument; }

public void setInstrument(Instrument instrument) {this.instrument = instrument; }

@Override

public void perform()

{

try

{

System.*out*.print("Playing " + song + " : ");

instrument.play();

}

catch(Throwable e)

{

e.printStackTrace();

}

}}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Performer.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.aop.bean;

public interface Performer

{

public void perform();

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Piano.java\*\*\*\*\*\*\*\*\*\*\***

package com.cg.aop.bean;

public class Piano implements Instrument

{

@Override

public void play()

{

System.*out*.println(" PLINK Plink Plink");

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Tabala.java\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

package com.cg.aop.bean;

public class Tabla implements Instrument

{

@Override

public void play()

{

System.*out*.println(" Tap Tap Tap");

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*schemaaop.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:aop=*"http://www.springframework.org/schema/aop"*

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

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

*http://www.springframework.org/schema/aop*

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

<aop:aspectj-autoproxy />

<bean id=*"asp"* class=*"com.cg.advices.AudienceAspect"* >

<property name=*"audience"* ref=*"aud"*/>

</bean>

<bean id=*"aud"* class=*"com.cg.aop.bean.Audience"* />

<bean id=*"tabala"* class=*"com.cg.aop.bean.Tabla"* />

<bean id=*"zakirHusein"* class=*"com.cg.aop.bean.Instrumentalist"*>

<property name=*"instrument"* ref=*"tabala"*/>

<property name=*"song"* value=*"Ja Ja Re Apane "*/>

</bean>

<aop:config>

<aop:aspect ref=*"asp"*>

<aop:pointcut id=*"mypoint"*

expression=*"execution(\* com.cg.aop.bean.Performer.perform())"*/>

<aop:around

pointcut=*"execution(\* com.cg.aop.bean.Performer.perform())"*

method=*"performAround"* />

<aop:before

pointcut-ref=*"mypoint"*

method=*"performBefore"* />

<aop:after

pointcut-ref=*"mypoint"*

method=*"performAfter"* />

<aop:after-returning

pointcut=*"execution(\* com.cg.aop.bean.Performer.perform())"*

method=*"performAfterThrowing"* />

</aop:aspect>

</aop:config>

</beans>

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*project/src/userprop.properties\*\*\*\*\*\*\*\*\*\***

username=CG

password=CG

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Thank You\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***