C22058 **January 27, 2023**

Java Spring Laboratory 3:

Demonstrating Constructor Injection of a Collection type using XML based Wiring of Spring Application.

Program 1 – Write a program to demonstrate constructor injection of a list of string literals by modeling the below entities: A Rich Person owns a list of cars. Each car is described as a String literal. Assume that every Rich Person has a first name and a last name in addition to a list of owned car's description. All these three parameters should be accepted as constructor arguments. Use XML based configuration to wire your application. Define Person type as an interface and RichPerson type as a class that implements Person interface. Person interface defines a method named as aboutMyself which when invoked on a concrete Person type should simply print the information about that person which includes the person's full name and cars owned by that person.

Solution:

Person.java

```java
package nmitd.spring;

public interface Person {

       public void aboutMyself();
}
```

RichPerson.java

```java
package nmitd.spring;
import java.util.List;
public class RichPerson implements Person {

       private String firstName;
       private String lastName;
       private List<String> carsOwned;

       public RichPerson(String firstName, String lastName, List<String> 
carsOwned) {
              this.firstName=firstName;
              this.lastName=lastName;
              this.carsOwned=carsOwned;
       }

       public void aboutMyself()
       {
              System.out.println("Myself "+firstName+" "+lastName+" am a rich 
person. I own these cars:");
              for(String myCar:carsOwned)
              {
                     System.out.println("-Cars: "+myCar);
              }
       }
}
```

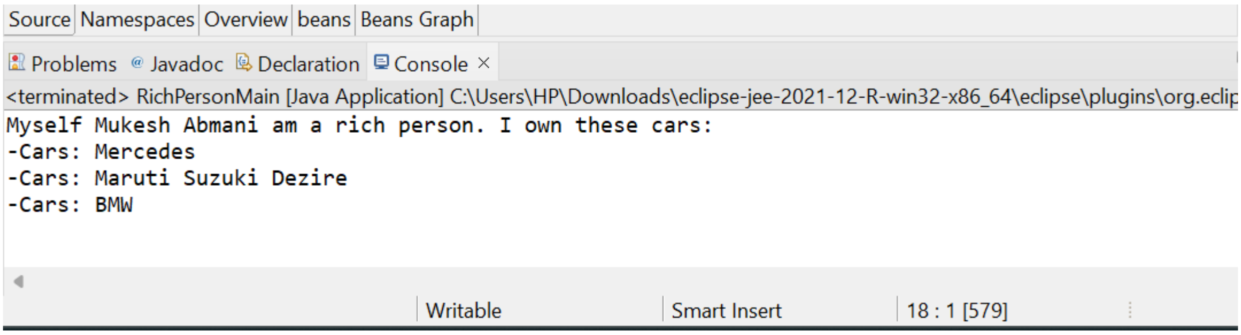
RichPersonMain.java

```
package nmitd.spring;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class RichPersonMain {

        public static void main(String[] args) throws Exception
        {
                ClassPathXmlApplicationContext context=new
ClassPathXmlApplicationContext("nmitd/spring/myBeans.xml");
                Person p1=context.getBean(Person.class);
                p1.aboutMyself();
                context.close();
        }
}
```

myBeans.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="p1" class="nmitd.spring.RichPerson">
        <constructor-arg value="Mukesh"/>
        <constructor-arg value="Abmani"/>
        <constructor-arg>
                <list>
                        <value>Mercedes</value>
                        <value>Maruti Suzuki Dezire</value>
                        <value>BMW</value>
                </list>
        </constructor-arg>
</bean>
</beans>
```

Output:



```
<terminated> RichPersonMain [Java Application] C:\Users\HP\Downloads\eclipse-jee-2021-12-R-win32-x86_64\eclipse\plugins\org.eclip
Myself Mukesh Abmani am a rich person. I own these cars:
-Cars: Mercedes
-Cars: Maruti Suzuki Dezire
-Cars: BMW
```

Program 2 – Modify program 1 such that instead of modeling each car as a String type model it using your own user defined Car type. Define Car as an Interface. And define some concrete Car class; For instance, Mercedes can be defined as a class implementing the Car interface. Every concrete Car class should maintain information about the model and manufacturing year of that car. The constructor of the class should accept these as String parameters. The Car interface specifies a method drive the implementation of which simply prints information about the actual car (its make, model, year etc.) which is being driven. Thus, in this program we want constructor injection of a list of Car types into an object of RichPerson type.

Person.java

```
package nmitd.spring;

public interface Person {

        public void aboutMyself();
}
```

RichPerson.java

```
package nmitd.spring;
import java.util.List;
public class RichPerson implements Person {

        //private Cars car;
        private String firstName;
        private String lastName;
        private List<Car> carsOwned;

        public RichPerson(String firstName,String lastName,List<Car> carsOwned) {
                this.firstName=firstName;
                this.lastName=lastName;
                this.carsOwned=carsOwned;
        }

        public void aboutMyself()
        {
                System.out.println("Myself "+firstName+" "+lastName+" am a rich
person. I own these cars:");
                for(Car myCar:carsOwned)
                {
                        myCar.drive();
                }

        }
}
```

Car.java

```
package nmitd.spring;

public interface Car {

        public void drive();
}
```

Mercedes.java

```
package nmitd.spring;

import java.io.PrintStream;

public class Mercedes implements Car {
        private PrintStream stream;

        public Mercedes(PrintStream stream)
        {
                this.stream=stream;
        }
        @Override
        public void drive() {
                // TODO Auto-generated method stub
                stream.println("Mercedes 2008 ");
        }

}
```

Maruti.java

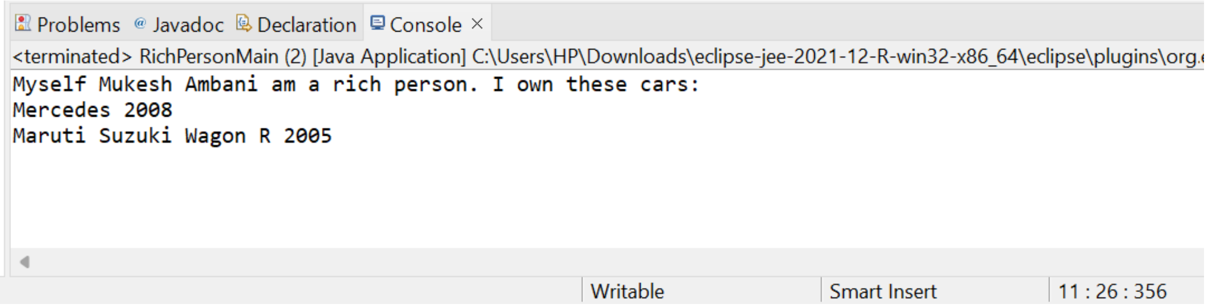
```
package nmitd.spring;
import java.io.PrintStream;
public class Maruti implements Car {

        PrintStream stream;
        public Maruti(PrintStream stream) {
                this.stream=stream;
        }
        @Override
        public void drive() {
                // TODO Auto-generated method stub
                stream.println("Maruti Suzuki Wagon R 2005");
        }

}
```

RichPersonMain.java

```
package nmitd.spring;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class RichPersonMain {

        public static void main(String[] args)
        {
                ClassPathXmlApplicationContext context=new
ClassPathXmlApplicationContext("nmitd/spring/myBeans.xml");
                Person p1=context.getBean("p1",Person.class);
                p1.aboutMyself();
                context.close();
        }
}
```

myBeans.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="p1" class="nmitd.spring.RichPerson">
       <constructor-arg value="Mukesh"/>
       <constructor-arg value="Ambani"/>
       <constructor-arg>
              <list>
                     <ref bean="car1"/>
                     <ref bean="car2"/>
              </list>
       </constructor-arg>
</bean>

<bean id="car1" class="nmitd.spring.Mercedes">
       <constructor-arg value="#{T(System).out}"/>
</bean>

<bean id="car2" class="nmitd.spring.Maruti">
       <constructor-arg value="#{T(System).out}"/>
</bean>
</beans>
```



```
<terminated> RichPersonMain (2) [Java Application] C:\Users\HP\Downloads\eclipse-jee-2021-12-R-win32-x86_64\eclipse\plugins\org.
Myself Mukesh Ambani am a rich person. I own these cars:
Mercedes 2008
Maruti Suzuki Wagon R 2005
```