

Practical-3

Aim : Develop application using Spring Framework, Lightweight Containers and Dependency Injection with Spring.

Theory:

Autowiring Modes XML		
No.	Mode	Description
1)	no	It is the default autowiring mode. It means no autowiring by default.
2)	byName	The byName mode injects the object dependency according to name of the bean. In such case, property name and bean name must be same. It internally calls setter method.
3)	byType	The byType mode injects the object dependency according to type. So property name and bean name can be different. It internally calls setter method.
4)	constructor	The constructor mode injects the dependency by calling the constructor of the class. It calls the constructor having large number of parameters.
5)	autodetect	It is deprecated since Spring 3.

The `@Autowired` annotation marks a Constructor, Setter method, Properties and Config() method as to be autowired that is 'injecting beans'(Objects) at runtime by Spring Dependency Injection.

After enabling `@Autowired` annotation and deciding on what configuration to be used. The beans can be wired via constructor or properties or setter method.

Constructor injection

We can inject the dependency by constructor. The `<constructor-arg>` sub-element of `<bean>` is used for constructor injection. Here we are going to inject

- 1} primitive and String-based values
- 2} Dependent object (contained object)
- 3} Collection values etc.

Setter Injection

We can inject the dependency by setter method also. The `<property>` subelement of `<bean>` is used for setter injection. Here we are going to inject

- 1} primitive and String-based values
- 2} Dependent object (contained object)
- 3} Collection values etc.

Q1} Write a program to create Student class (name, Department) and Department (departmentName, Departmentid, Major) class Student Has-A relationship with Department class. Use constructor injection to invoke object of Department class in Student class.

Code:

Student.java

```
package com.student.ci;

public class Student {

    private String name;

    private String dept;

    private Department depart;

    Student(String name,String dept, Department depart)

    {

        this.name = name;

        this.dept = dept;

        this.depart = depart;

    }

    public String toString() {

        return "Student [name=" + name + ", dept=" + dept + ", depart=" + depart + "];

    }

}
```

Department.java

```
package com.student.ci;

public class Department {

    int deptid;

    String deptname;

    String major;

    public Department(int deptid, String deptname, String major) {

        super();

        this.deptid = deptid;

        this.deptname = deptname;
```

```
this.major = major;

}

public String toString() {

// TODO Auto-generated method stub

return this.deptid + " "+this.deptname+" "+this.major;

}

}
```

Test.java

```
package com.student.ci;

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

public class Test {

    public static void main(String[] args) {

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

        Student s = (Student) context.getBean("student");

        System.out.print(s);

    }

}
```

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

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

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

<bean class="com.student.ci.Department" name="dep">

<constructor-arg value="1"/>
```

```
<constructor-arg value="COMPS"/>

<constructor-arg value="AI"/>

</bean>

<bean class="com.student.ci.Student" name="student">

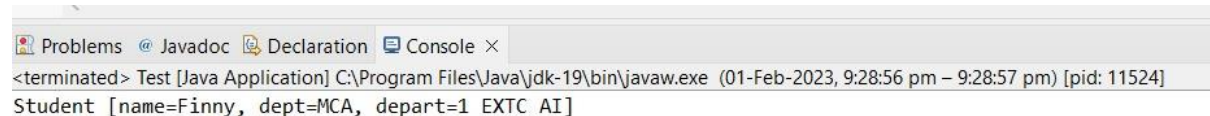
<constructor-arg value="Sunny"/>

<constructor-arg value="MCA"/>

<constructor-arg ref="dep" />

</bean>

</beans>
```

Output:

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (01-Feb-2023, 9:28:56 pm - 9:28:57 pm) [pid: 11524]
Student [name=Finny, dept=MCA, depart=1 EXTC AI]
```

Q2} Write a program create Student class and Department class. Student class Has-A relationship with Department class. Use setter injection to invoke object of Department class in Student class.

Code:**Student.java**

```
package com.student.si;

public class Student {

private String name;

private String dept;

private Department depart;

Student(String name,String dept, Department depart)

{

this.name = name;

this.dept = dept;

this.depart = depart;
```

```
}

public Student() {

    super();

    // TODO Auto-generated constructor stub

}

public String getName() {

    return name;

}

public void setName(String name) {

    this.name = name;

}

public String getDept() {

    return dept;

}

public void setDept(String dept) {

    this.dept = dept;

}

public Department getDepart() {

    return depart;

}

public void setDepart(Department depart) {

    this.depart = depart;

}

public String toString() {

    return "Student [name=" + name + ", dept=" + dept + ", depart=" + depart + "];"

}

}
```

Department.java

```
package com.student.si;

public class Department {

    int deptid;

    String deptname;

    String major;

    public Department() {

        super();

        // TODO Auto-generated constructor stub

    }

    public int getDeptid() {

        return deptid;

    }

    public void setDeptid(int deptid) {

        this.deptid = deptid;

    }

    public String getDeptname() {

        return deptname;

    }

    public void setDeptname(String deptname) {

        this.deptname = deptname;

    }

    public String getMajor() {

        return major;

    }

    public void setMajor(String major) {

        this.major = major;

    }

}
```

```
}

public Department(int deptid, String deptname, String major) {

    super();

    this.deptid = deptid;

    this.deptname = deptname;

    this.major = major;

}

public String toString() {

    // TODO Auto-generated method stub

    return this.deptid + " " + this.deptname + " " + this.major;

}

}
```

Test.java

```
package com.student.si;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
public class Test {
    public static void main(String[] args) {

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

        Student s = (Student) context.getBean("student");
        System.out.print(s);}
}
```

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

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

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

<bean class="com.student.si.Department" name="dep">
```

```
<property name="deptid" value="101"/>

<property name="deptname" value="EXTC"/>

<property name="major" value="AI"/>

</bean>

<bean class="com.student.si.Student" name="student">

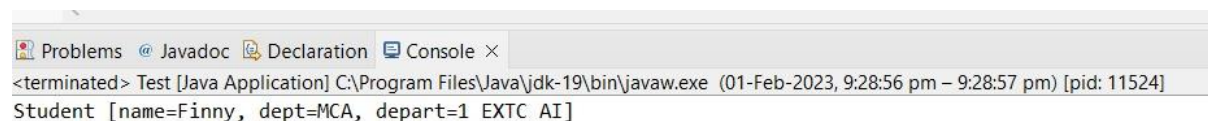
<property name="name" value="Sunny"/>

<property name="dept" value="MCA"/>

<property name="depart" ref="dep" />

</bean>

</beans>
```

Output:

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (01-Feb-2023, 9:28:56 pm – 9:28:57 pm) [pid: 11524]
Student [name=Finny, dept=MCA, depart=1 EXTC AI]
```

Q3} Write a program create Car class (CarColor, Brand) and Engine (Chesiss Number, Gears) class. Car Has-A relationship with Engine class. Use Auto-wiring through XML (any one byname/byType/constructor) for dependency injection.

Code:**Car.java**

```
package com.cars.auto.wire;

public class Car {

    private Engine engine;

    private String carcolor;

    private String brand;

    public Engine getEngine() {

    return engine;

    }
```



```
}

public void setEngine(Engine engine) {

    this.engine = engine;

}

public String getCarcolor() {

    return carcolor;

}

public void setCarcolor(String carcolor) {

    this.carcolor = carcolor;

}

public String getBrand() {

    return brand;

}

public void setBrand(String brand) {

    this.brand = brand;

}

public Car() {

    super();

    // TODO Auto-generated constructor stub

}

public Car(Engine engine, String carcolor, String brand) {

    super();

    this.engine = engine;

    this.carcolor = carcolor;

    this.brand = brand;

}

public String toString() {
```

```
return "Car [engine=" + engine + ", carcolor=" + carcolor + ", brand=" + brand +
    "];"
}
}
```

Engine.java

```
package com.cars.auto.wire;

public class Engine {

    private int chassisno;

    private int gears;

    public int getChassisno() {

        return chassisno;

    }

    public void setChassisno(int chassisno) {

        this.chassisno = chassisno;

    }

    public int getGears() {

        return gears;

    }

    public void setGears(int gears) {

        this.gears = gears;

    }

    public String toString() {

        return "Engine [chassisno=" + chassisno + ", gears=" + gears + "];"

    }

}
```

Test.java

```
package com.cars.auto.wire;

import org.springframework.context.ApplicationContext;
```

```
import org.springframework.context.support.ClassPathXmlApplicationContext;
import com.springcore.auto.wire.Emp;
```

```
public class Test {
    public static void main(String[] args) {
        ApplicationContext context = new
ClassPathXmlApplicationContext("config2.xml");
        Car car1 = (Car) context.getBean("car1",Car.class);
        System.out.print(car1);
    }
}
```

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

<bean class="com.cars.auto.wire.Engine" name="engine">

<property name="chassisno" value="1234" />

<property name="gears" value="5" />

</bean>

<bean class="com.cars.auto.wire.Car" name="car1" autowire="byType" >

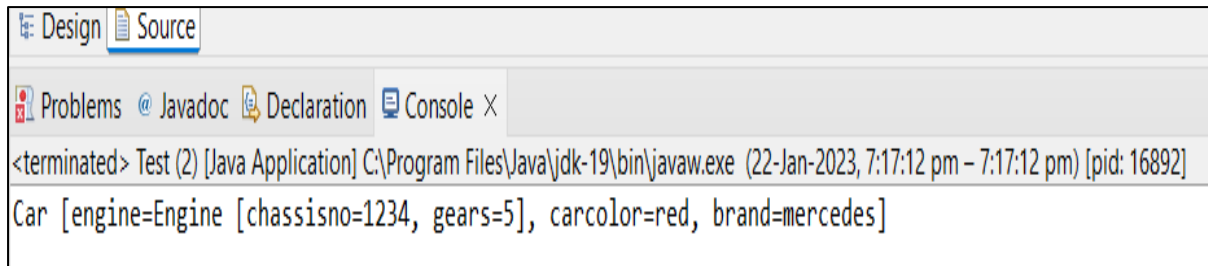
<property name="carcolor" value="red" />

<property name="brand" value="mercedes" />

</bean>

</beans>
```

Output:



Q4) Write a program create Car class and Engine class. Car Has-A relationship with Engine class. Use @autowiring to invoke dependency injection (any two method).

Code:

Car.java

1} Autowiring by Property

```
package com.cars.auto.wire.annotation;

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

public class Car {

    @Autowired

    private Engine engine;

    private String carcolor;

    private String brand;

    public Engine getEngine() {

        return engine;

    }

    public void setEngine(Engine engine) {

        this.engine = engine;

    }

    public String getCarcolor() {

        return carcolor;

    }

    public void setCarcolor(String carcolor) {

        this.carcolor = carcolor;

    }

}
```

```
public String getBrand() {  
    return brand;  
}  
  
public void setBrand(String brand) {  
    this.brand = brand;  
}  
  
public Car() {  
    super();  
    // TODO Auto-generated constructor stub  
}  
  
public Car(Engine engine, String carcolor, String brand) {  
    super();  
    this.engine = engine;  
    this.carcolor = carcolor;  
    this.brand = brand;  
}  
  
public String toString() {  
    return "Car [engine=" + engine + ", carcolor=" + carcolor + ", brand=" + brand +  
    "];"  
}  
}
```

2} Autowiring by Setter Method

```
package com.cars.auto.wire.annotation;  
  
import org.springframework.beans.factory.annotation.Autowired;  
  
public class Car {  
    private Engine engine;  
    private String carcolor;  
    private String brand;
```

```
public Engine getEngine() {  
    return engine;  
}  
  
@Autowired  
public void setEngine(Engine engine) {  
    this.engine = engine;  
}  
  
public String getCarcolor() {  
    return carcolor;  
}  
  
public void setCarcolor(String carcolor) {  
    this.carcolor = carcolor;  
}  
  
public String getBrand() {  
    return brand;  
}  
  
public void setBrand(String brand) {  
    this.brand = brand;  
}  
  
public Car() {  
    super();  
    // TODO Auto-generated constructor stub  
}  
  
public Car(Engine engine, String carcolor, String brand) {  
    super();  
    this.engine = engine;  
    this.carcolor = carcolor;  
}
```

```
this.brand = brand;

}

public String toString() {

return "Car [engine=" + engine + ", carcolor=" + carcolor + ", brand=" + brand +
    "]\n";

}

}
```

Engine.java

```
package com.cars.auto.wire.annotation;

public class Engine {

private int chassisno;

private int gears;

public int getChassisno() {

return chassisno;

}

public void setChassisno(int chassisno) {

this.chassisno = chassisno;

}

public int getGears() {

return gears;

}

public void setGears(int gears) {

this.gears = gears;

}

public String toString() {

return "Engine [chassisno=" + chassisno + ", gears=" + gears + "]\n";

}
```

```
}
```

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

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.xsd">

<context:annotation-config/>

<bean class="com.cars.auto.wire.annotation.Engine" name="engine">

<property name="chassisno" value="0007" />

<property name="gears" value="6" />

</bean>

<bean class="com.cars.auto.wire.annotation.Car" name="car1">

<property name="carcolor" value="yellow " />

<property name="brand" value="Lamborgihi " />

</bean>

</beans>
```

Test.java

```
package com.cars.auto.wire.annotation;

import org.springframework.context.ApplicationContext;

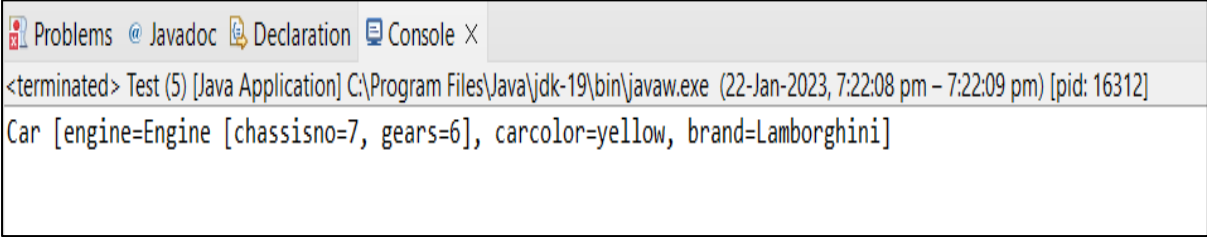
import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.springcore.auto.wire.Emp;

public class Test {
```



```
public static void main(String[] args) {  
  
    ApplicationContext context = new ClassPathXmlApplicationContext("configannot.xml");  
  
    Car car1 = (Car) context.getBean("car1",Car.class);  
  
    System.out.print(car1);  
  
}  
  
}
```

Output:

The screenshot shows an IDE console window with the following content:

```
<terminated> Test (5) [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (22-Jan-2023, 7:22:08 pm - 7:22:09 pm) [pid: 16312]  
Car [engine=Engine [chassisno=7, gears=6], carcolor=yellow, brand=Lamborghini]
```

Conclusion:- I have successfully learned Spring Framework, Lightweight Containers and Dependency Injection with Spring.