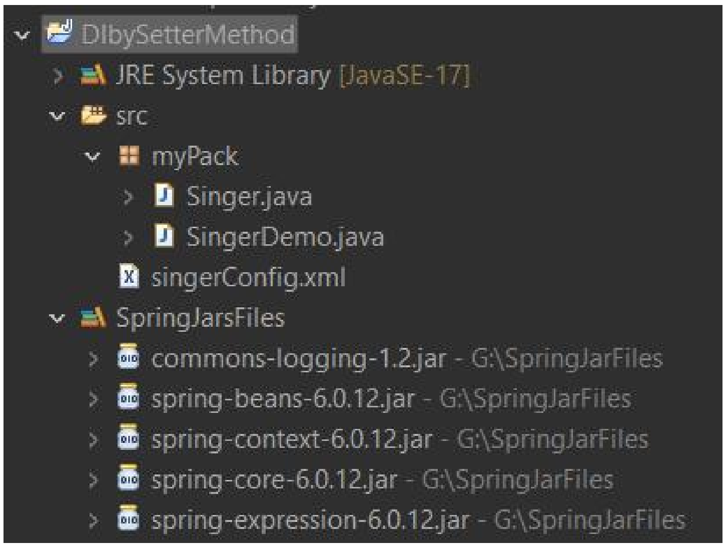
PRACTICAL NO. 03
Introduction to Spring Framework

LOB 3 Demonstrate Data Access with Spring framework.
LO3 Develop application using Spring Framework, Lightweight Containers and Dependency Injection.

SET 1: Write a program to demonstrate dependency injection via setter method.

JAR Files:



```
DIbySetterMethod
  JRE System Library [JavaSE-17]
  src
    myPack
      Singer.java
      SingerDemo.java
    singerConfig.xml
  SpringJarsFiles
    commons-logging-1.2.jar - G:\SpringJarFiles
    spring-beans-6.0.12.jar - G:\SpringJarFiles
    spring-context-6.0.12.jar - G:\SpringJarFiles
    spring-core-6.0.12.jar - G:\SpringJarFiles
    spring-expression-6.0.12.jar - G:\SpringJarFiles
```

Code:
Singer.java

```java
package myPack;

public class Singer {
	int age;
	String name;
	public int getAge() {
		return age;
	}
	public void setAge(int age) {
		this.age = age;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public Singer(int age, String name) {
		super();
		this.age = age;
		this.name = name;
	}
	public Singer() {
		super();
	}
	@Override
```

```
        public String toString() {
                return "Singer [age=" + age + ", name=" + name + "]";
        }
}
```

singerConfig.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-3.0.xsd">
<bean id="singBean" class="myPack.Singer">
<property name="name" value="Onkar"></property>
<property name="age" value="23"></property>
</bean>
<bean id="singBean1" class="myPack.Singer">
<property name="name" value="Raju"></property>
<property name="age" value="20"></property>
</bean>
</beans>
```

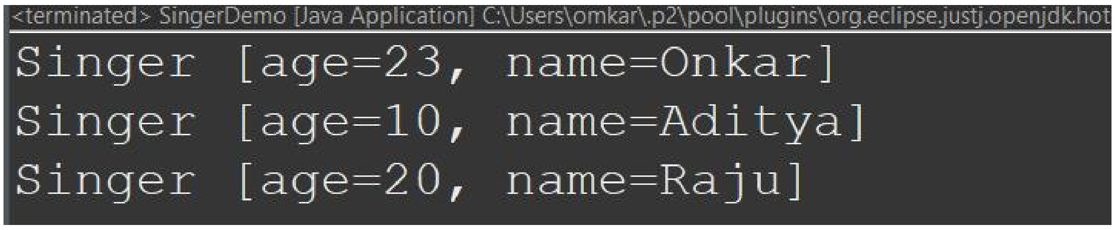
SingerDemo.java

```
package myPack;

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

public class SingerDemo {
        private static ApplicationContext ctx;
        public static void main(String[] args) {
                // TODO Auto-generated method stub
                ctx=new
        ClassPathXmlApplicationContext("singerConfig.xml");
                Singer s1 = (Singer) ctx.getBean("singBean");
                System.out.println(s1);
                s1.setAge(10);
                s1.setName("Aditya");
                System.out.println(s1);
                s1 = (Singer) ctx.getBean("singBean1");
                System.out.println(s1);
        }
}
```

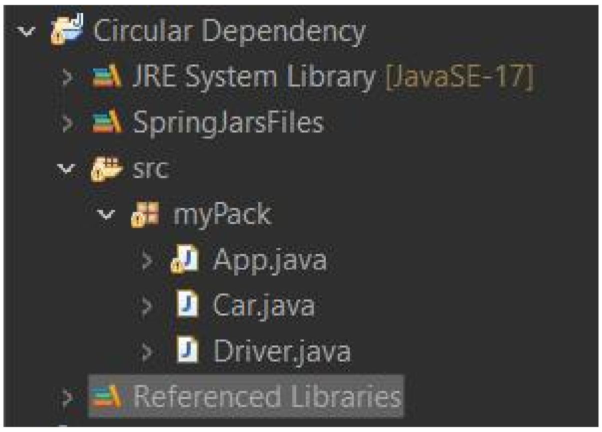
Output:



```
<terminated> SingerDemo [Java Application] C:\Users\omkar\.p2\pool\plugins\org.eclipse.justj.openjdk.hot
Singer [age=23, name=Onkar]
Singer [age=10, name=Aditya]
Singer [age=20, name=Raju]
```

SET 2: Write a program to demonstrate circular dependency.

JAR Files:



Code:

Car.java

```
package myPack;

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

@Component
public class Car {
	private Driver driverObj;

	@Autowired
	Car(Driver driverObj) {
		super();
		this.driverObj = driverObj;
	}

	public Driver getDriverObj() {
		return driverObj;
	}

	void setDriverObj(Driver driverObj) {
		this.driverObj = driverObj;
	}

	@Override
	public String toString() {
		return "Car [driverObj=" + driverObj + "]";
	}
}
```

Driver.java

```
package myPack;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Lazy;
import org.springframework.stereotype.Component;

@Component
```

```
public class Driver {
	private Car carObj;

	@Autowired
	Driver(@Lazy Car carObj) {
		super();
		this.carObj = carObj;
	}

	public Car getCarObj() {
		return carObj;
	}

	public void setCarObj(Car carObj) {
		this.carObj = carObj;
	}

	@Override
	public String toString() {
		return "Driver [carObj=" + carObj + "]";
	}
}
```

App.java

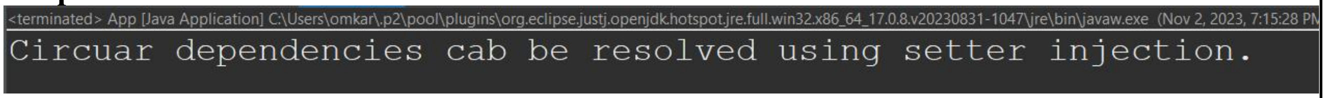
```
package myPack;

import org.springframework.context.annotation.AnnotationConfigApplicationContext;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.support.AbstractApplicationContext;

@Configuration
@ComponentScan(basePackages="myPack")
public class App {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		AbstractApplicationContext ctx=new
	AnnotationConfigApplicationContext(App.class);
		System.out.println("Circuar dependencies cab be resolved using setter
	injection.");
	}
}
```

Output:



```
<terminated> App [Java Application] C:\Users\omkar\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (Nov 2, 2023, 7:15:28 PM
Circuar dependencies cab be resolved using setter injection.
```