Assignment No. 7

Spring Framework

1. Write a program to print “Hello World” using spring framework.

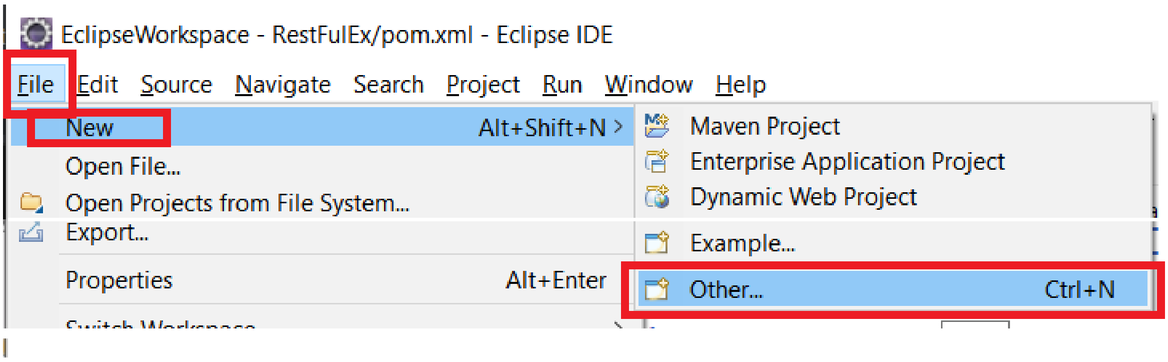
2. Write a program to demonstrate dependency injection via setter method.

3. Write a program to demonstrate dependency injection via Constructor.

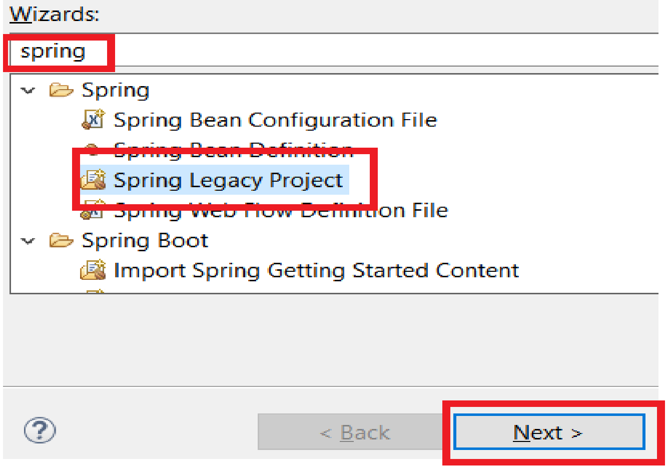
Steps to Create Spring Legacy Project

Step 1 : Creating Spring Legacy Project.

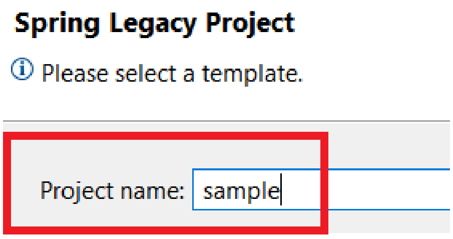
1.1 : Open Eclipse. Go To File > New > Other.



1.2 : Search for ‘spring’ and Select ‘Spring Legacy Project’. Then Click on Next.

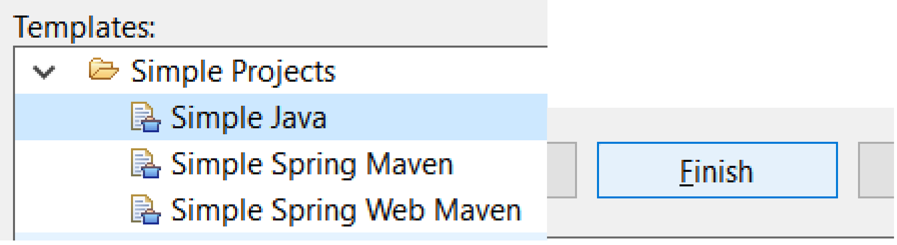

Wizards: spring
Spring
Spring Bean Configuration File
Spring Bean Definition
Spring Legacy Project
Spring Web Flow Definition File
Spring Boot
Import Spring Getting Started Content
< Back
Next >

1.3 : ChooseProject Name of your wish, below there select Simple Java & simply Finish.

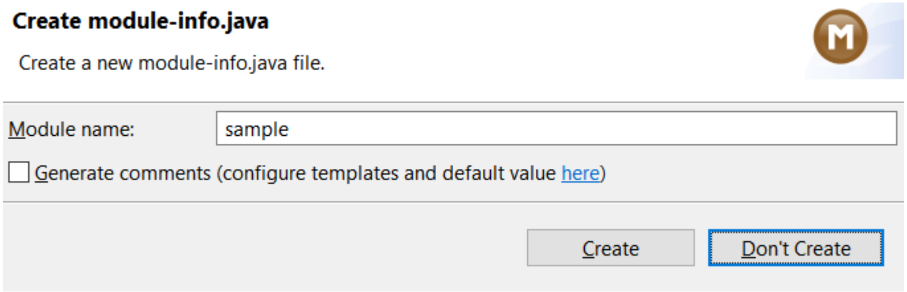

Spring Legacy Project

Please select a template.

Project name: sample



1.4 : If asked for Creating module-info.java file, click on Don’t Create.


Create module-info.java

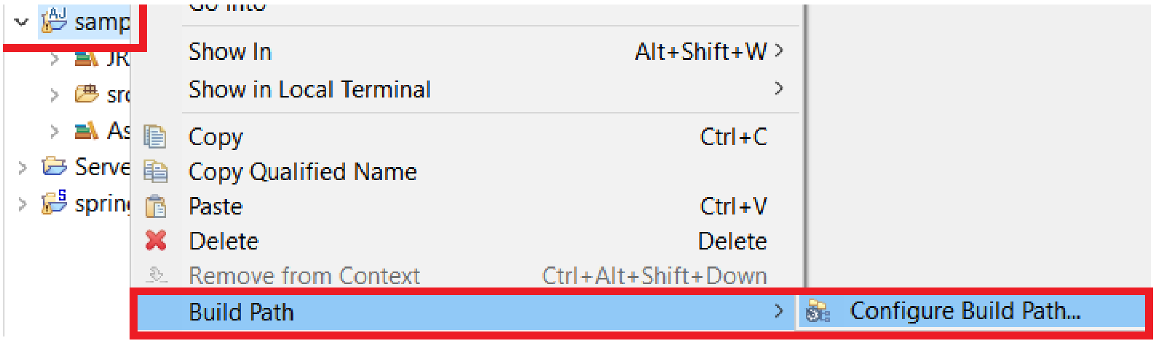
Create a new module-info.java file.

Module name: sample

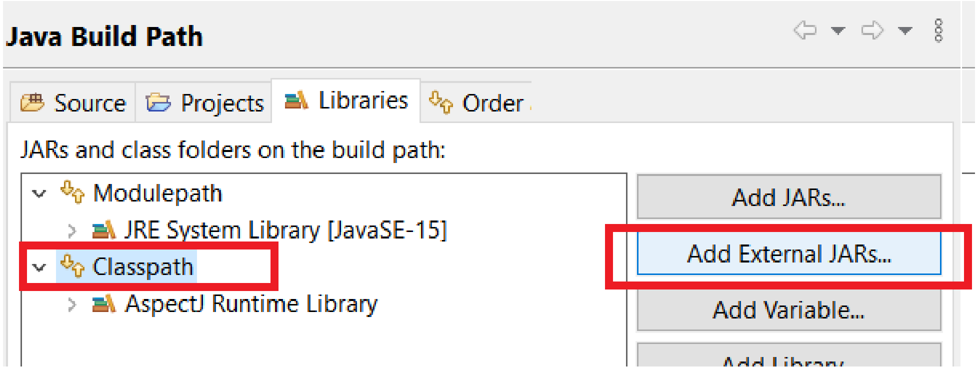
☐ Generate comments (configure templates and default value here)

Step 2 : Adding the Spring Libraries.

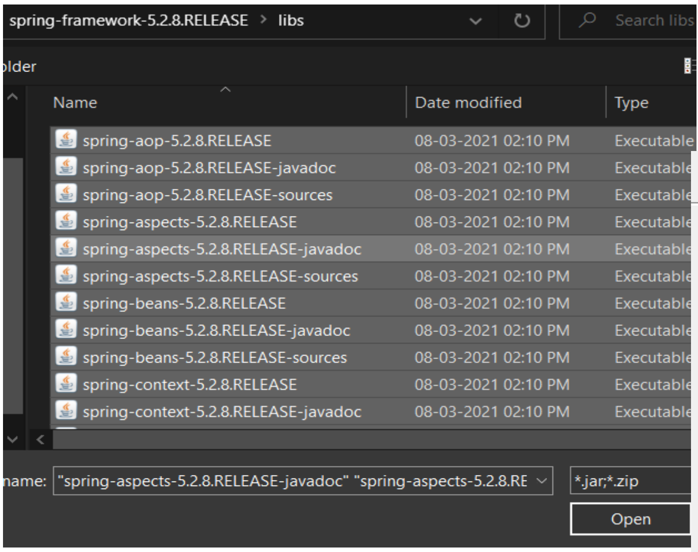
2.1 : Right click on your Newly created Spring Legacy project, Choose Build Path > Configure Build Path.



2.2 On Java Build Path wizard, Choose Classpath and then select Add External JARs.

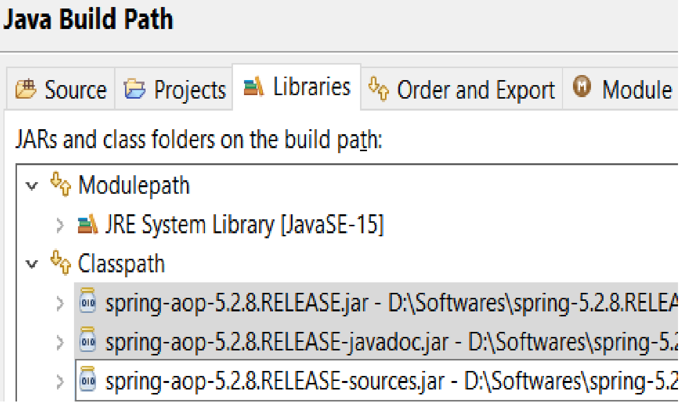


2.3 : Choose all the Spring Libraries you've downloaded, and click on OPEN. This will add all libraries to Classpath.



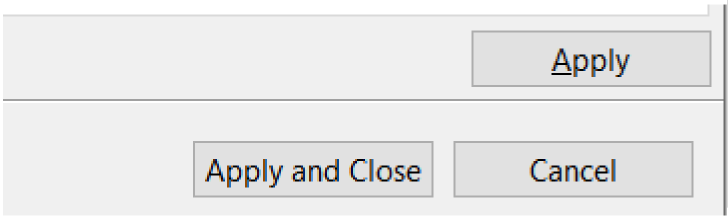
Name	Date modified	Type
spring-aop-5.2.8.RELEASE	08-03-2021 02:10 PM	Executable
spring-aop-5.2.8.RELEASE-javadoc	08-03-2021 02:10 PM	Executable
spring-aop-5.2.8.RELEASE-sources	08-03-2021 02:10 PM	Executable
spring-aspects-5.2.8.RELEASE	08-03-2021 02:10 PM	Executable
spring-aspects-5.2.8.RELEASE-javadoc	08-03-2021 02:10 PM	Executable
spring-aspects-5.2.8.RELEASE-sources	08-03-2021 02:10 PM	Executable
spring-beans-5.2.8.RELEASE	08-03-2021 02:10 PM	Executable
spring-beans-5.2.8.RELEASE-javadoc	08-03-2021 02:10 PM	Executable
spring-beans-5.2.8.RELEASE-sources	08-03-2021 02:10 PM	Executable
spring-context-5.2.8.RELEASE	08-03-2021 02:10 PM	Executable
spring-context-5.2.8.RELEASE-javadoc	08-03-2021 02:10 PM	Executable

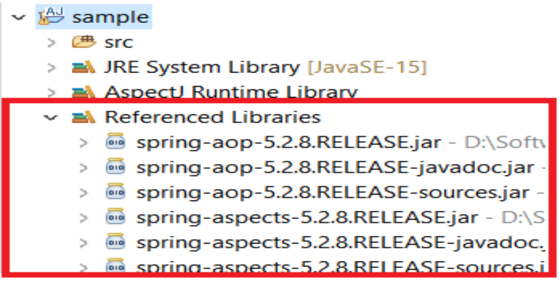
name: "spring-aspects-5.2.8.RELEASE-javadoc" "spring-aspects-5.2.8.RE *.jar;*.zip



spring-aop-5.2.8.RELEASE.jar - D:\Softwares\spring-5.2.8.RELEA
spring-aop-5.2.8.RELEASE-javadoc.jar - D:\Softwares\spring-5.2
spring-aop-5.2.8.RELEASE-sources.jar - D:\Softwares\spring-5.2

2.4 Finally click on Apply & Close, now you are ready to work with Spring Legacy Project.





spring-aop-5.2.8.RELEASE.jar - D:\Softw
spring-aop-5.2.8.RELEASE-javadoc.jar -
spring-aop-5.2.8.RELEASE-sources.jar -
spring-aspects-5.2.8.RELEASE.jar - D:\S
spring-aspects-5.2.8.RELEASE-javadoc.
spring-aspects-5.2.8.RELEASE-sources.j

Problem Statement 1 : Write a program to print "Hello World" using spring framework.

Solution :

HelloWorld.java

```
package spring1;

public class HelloWorld {

	String name;

	public String getName() {
		return name;
	}

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

	@Override
	public String toString() {
		return "Hello World, I'm " + name + ".";
	}
}
```

appctx3.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="hw" class="spring1.HelloWorld">
		<property name="name" value="Nidhi"/>
	</bean>

</beans>
```

TestHelloWorld.java

```
package spring1;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class TestHelloWorld {

	public static void main(String[] args) {

		ClassPathXmlApplicationContext app = new 
ClassPathXmlApplicationContext("appctx3.xml");
		HelloWorld hw = (HelloWorld) app.getBean("hw");

		System.out.println(hw.toString());

	}

}
```

Output :

```
Hello World, I’m Nidhi
```

Problem Statement 2 : Write a program to demonstrate dependency injection via Constructor.

Solution:

Address.java

```
package depinjectionbycons;

public class Address {

	private String city;
	private String state;
	private String country;

	public Address(String city, String state, String country) {
		super();
		this.city = city;
		this.state = state;
		this.country = country;
	}

	public String toString(){
		return city+" "+state+" "+country;
	}

}
```

Employee.java

```
package depinjectionbycons;



public class Employee {

	private int id;
	private String name;
	private Address address;

	public Employee() {System.out.println("def cons");}


	public Employee(int id, String name, Address address) {
		super();
		this.id = id;
		this.name = name;
		this.address = address;
	}

	void show(){
		System.out.println(id+" "+name);
		System.out.println(address.toString());
	}

}
```

applicationContext.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 id="e" class="depinjectionbycons.Employee"></bean>
-->

<!--<bean id="e" class="depinjectionbycons.Employee">
<constructor-arg value="10" type="int"></constructor-arg>
</bean>
-->

<!--<bean id="e" class="depinjectionbycons.Employee">
<constructor-arg value="10"></constructor-arg>
</bean>
-->
<!--<bean id="e" class="depinjectionbycons.Employee">
<constructor-arg value="nidhi"></constructor-arg>
</bean>
-->

<!--<bean id="e" class="depinjectionbycons.Employee">
<constructor-arg value="12" type="int"></constructor-arg>
<constructor-arg value="nidhi"></constructor-arg>
</bean>
-->
<bean id="a1" class="depinjectionbycons.Address">
<constructor-arg value="nerul"></constructor-arg>
<constructor-arg value="mumbai"></constructor-arg>
<constructor-arg value="India"></constructor-arg>
</bean>


<bean id="e" class="depinjectionbycons.Employee">
<constructor-arg value="12" type="int"></constructor-arg>
<constructor-arg value="nidhi"></constructor-arg>
<constructor-arg>
<ref bean="a1"/>
</constructor-arg>
</bean>

</beans>
```

Test.java

```
package depinjectionbycons;
```

```
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;


public class Test {

        public static void main(String[] args) {
                Resource r=new ClassPathResource("applicationContext.xml");
                @SuppressWarnings("deprecation")
                BeanFactory factory=new ~~XmlBeanFactory~~(r);

                Employee s=(Employee)factory.getBean("e");
                s.show();

        }

}
```

Output :

```
12 nidhi
nerul mumbai India
```

Problem Statement 3 :

Write a program to demonstrate dependency injection via setter method.

Solution:

Singer.java

```
package spring1;

public class Singer {
        String name;
        int age;
        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;
        }
void displayInfo()
{
        System.out.println("Name:" +name+" Age:" +age);
}
}
```

appctx.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="Singer" class="spring1.Singer">
<property name="name" value="shanaya"></property>
<property name="age" value="11"></property>
</bean>

</beans>
```

SingerTest.java

```
package spring1;

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

public class SingerTest {

	private static ApplicationContext ctx;
	public static void main(String[] args) {
		// TODO Auto-generated method stub
		ctx=new ClassPathXmlApplicationContext("appctx.xml");
		Singer singer=(Singer)ctx.getBean("Singer");
			singer.displayInfo();



	}

}
```

Output :

```
Name: Shanaya Age:11
```