# DAY-2

Task[1] → Tax Application

## Dependency Injection

- It is a kind of design pattern who's main task is to "inject the dependency" means inject or add one object into another object.
- used to achieve IOC.

Example:

```
class Student{
private String name;
private int roll_no;
private Address address;
}
```

The above class Student.java is dependent on another class i.e.,

```
class Address{
private int house_no;
private int pincode;
}
```

In the above example the Address.java class obj is being injected to the Student.java class.

it is used to achieve loose coupling in java.

**There are 2 ways via which you can add DI to the classes :**

## Setter Injection

** making use of the *ref* attribute as well as *property* tag

**Example:**

```
package in.sp.beans;
public class Student{

//defining the attributes
  private int roll_no;
  private String name;
  private Address address;

  //generating the setters

/*NOTE: It is necessary to have setters methods here.
          getter methods are optional*/

  public void setRollno(int rollno){
     this.roll_no = rollno;
  }


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


  public void setAddress(Address address){
     this.address = address;
  }

  //method to print the attributes
  public void display()
  {
```

```
    SOP("Roll_no:"+roll_no);
    SOP("Name:"+name);
    SOP("Address:"+address);
 }
}
```

```
package in.sp.beans;

public class Address
{
  private int houseno;
  private String city;
  private int pincode;

  /*NOTE: It is necessary to have setters methods here.
        getter methods are optional*/

  public void setHouseno(int houseno)
  {
     this.houseno = houseno;
  }

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

  public void setPincode(int pincode)
  {
     this.pincode= pincode
  }

  @Override
  public String toString()
```

```
  {
   return "#"+houseno+" ,"+city+" -"+pincode;
  }
```

```
<?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.springframewo

   <!-- bean definitions here -->

   <bean class = "in.sp.beans.Address" id = "addrId">
     <property name = "houseno" value = "111"/>
      <property name = "city" value = "delhi"/>
       <property name = "pincode" value = "123456"/>
   <bean/>

    <bean class = "in.sp.beans.Student" id = "stdId">
     <property name = "rollno" value = "101"/>
      <property name = "name" value = "Deepak"/>
       <property name = "address" ref = "addrId"/>
   <bean/>


</beans>
```

```
package in.sp.main;

public class Main
{
 public static void main(String[] args){
```

```
    ClassPathXmlApplicationContext context = new ClassPathXmlApplicationCon

      Student std = (Student) context.getBean(stdId);
      std.display();
  }
}
```

## Constructor Injection

** making use of the *ref* attribute as well as *constructor-arg* tag

**Example**:

```
package in.sp.beans;
public class Student{

//defining the attributes
  private int roll_no;
  private String name;
  private Address address;

 //constructor
 public Student(int rollno, String name, Address address))
 {
    this.roll_no = rollno;
    this.name = name;
    this.address = address;
 }

  //method to print the attributes
  public void display()
  {
```

```
    SOP("Roll_no:"+roll_no);
    SOP("Name:"+name);
    SOP("Address:"+address);
  }
}
```

```
package in.sp.beans;

public class Address
{
  private int houseno;
  private String city;
  private int pincode;

 public Address(int house, String city, int pincode)
 {
    this.houseno = house;
    this.city = city;
    this.pincode = pincode;
 }

  @Override
  public String toString()
  {
   return "#"+houseno+" ,"+city+" -"+pincode;
  }
```

```
<?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.springframewo
```

```
  <!-- bean definitions here -->

  <bean class = "in.sp.beans.Address" id = "addrId">
    <constructor-arg value = "222"/>
    <constructor-arg value = "delhi"/>
    <constructor-arg value = "123456"/>
  <bean/>

   <bean class = "in.sp.beans.Student" id = "stdId">
    <constructor-arg value = "102"/>
    <constructor-arg value = "deepak"/>
    <constructor-arg ref = "addrId"/>
  <bean/>

  </beans>
```

💡 While assigning values in the constructor write in the same order as mentioned in the class attributes else it throws error.

```
package in.sp.main;

public class Main
{
  public static void main(String[] args){
     ClassPathXmlApplicationContext context = new ClassPathXmlApplicationCon

        Student std = (Student) context.getBean(stdId);
        std.display();
  }
}
```

💡 The readability  and flexibility in case of setter DI is much good as compared to the constructor DI. Also