**Dependency Injection-**

The process of passing the required input from xml file to POJO class called as “Dependency injection.”

There are two types of dependency injection such as

* Setter base injection
* Constructor base injection

Note-interface base injection uses setter base injection internally.

**Setter base injection-**

By using setter method, we can achieve the setter base injection.

In this type of injection, the spring container uses setter method in the dependent (our) class for injecting its dependencies (primitive values. Or any).

Spring container knows whether to perform setter or constructor injection by reading the information from an external file called spring configuration file

In case of setter injection, the class must contain a setter method to get dependencies otherwise spring container doesn’t inject the dependencies to dependent object.

Example- using setter injection

**Student.java**

**package** com.test;

**public** **class** Student {

**private** String city;

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** **void** getMessage(String name) {

System.***out***.println("Hello,"+name+","+city);

}

}

**Test.java**

**package** com.test;

**import** org.springframework.context.ApplicationContext;

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

**public** **class** Test {

**public** **static** **void** main(String[] args) {

ApplicationContext ap= **new** ClassPathXmlApplicationContext("spring.xml");

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

student.getMessage("Velocity corporate training center");

}

}

**Spring.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-2.5.xsd"*>

<bean id=*"s"* class=*"com.test.Student"*>

<property name=*"city"* value=*"pune"*/>

</bean>

</beans>

**Pom.xml**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>spring</groupId>

<artifactId>SpringApplication</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

</dependencies>

<properties>

<spring.version>3.2.3.RELEASE</spring.version>

</properties>

</project>

**Output-**

Hello,Velocity corporate training center,pune

**Constructor base injection-**

By using parameterized constructor, we can achieve constructor base injection.

If spring container use parameterized constructor to create the spring bean class object, assign the values to spring bean properties called as constructor base injection.

**Student.java**

**package** com.test;

**public** **class** Student {

**private** String name;

**private** String age;

**private** String city;

**public** Student(String name, String age, String city) {

**super**();

**this**.name = name;

**this**.age = age;

**this**.city = city;

}

@Override

**public** String toString() {

**return** "Student [name=" + name + ", age=" + age + ", city=" + city + "]";

}

}

**Test.java**

**package** com.test;

**import** org.springframework.context.ApplicationContext;

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

**public** **class** Test {

**public** **static** **void** main(String[] args) {

ApplicationContext ap= **new** ClassPathXmlApplicationContext("spring.xml");

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

System.***out***.println(student);

}

}

**Spring.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-2.5.xsd"*>

<bean id=*"s"* class=*"com.test.Student"*>

<constructor-arg value=*"ashok"*></constructor-arg>

<constructor-arg value=*"25"*></constructor-arg>

<constructor-arg value=*"pune"*></constructor-arg>

</bean>

</beans>

**Pom.xml**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>spring</groupId>

<artifactId>SpringApplication</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

</dependencies>

<properties>

<spring.version>3.2.3.RELEASE</spring.version>

</properties>

</project>

**Output-**

Student [name=ashok, age=25, city=pune]