Bean lifecycle means, there are some methods is going to be call when the bean object will create and destroy.

There are 3 ways to manage bean lifecycle methods :

1. **Using XML**
2. **Using Annotation**
3. **By Implementing Interfaces**

The 3rd approach is least recommended i.e. **Implementing interface.** But we will still learn it for interview point of view.

Using Annotation

The **@PostConstruct** annotation use with that method which you want to be execute after creation of bean object and all the dependencies has been injected.

And the **@PreDestroy** annotation use with that method which you want to be execute before the bean object get destroy.

Before moving ahead,

Note that both **@PostConstruct** and **@PreDestroy** annotations are part of Java EE. And since Java EE has been deprecated in Java 9 and removed in Java 11 we have to add an additional dependency to use these annotations:

For Maven

<dependency>

<groupId>javax.annotation</groupId>

<artifactId>javax.annotation-api</artifactId>

<version>1.3.2</version>

</dependency>

**Person.java :**

package com.beanlifecycle.xml;

public class Person {

@PostConstruct

public void init() {

System.out.println("Entry - Take entry coin");

}

public void chilling() {

System.out.println("Consuming alcohol, dancing and watching beauties");

}

@PreDestroy

public void destroy() {

System.out.println("Exit - Give back entry coin");

}

}

**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"

. xsi:schemaLocation="

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

<context:annotation-config />

<bean id="person" class="com.beanlifecycle.xml.Person" />

</beans>

**Bar.java :**

public class Bar {

public static void main(String[] args) {

ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("com/beanlifecycle/xml/config.xml");

Person p = context.getBean("person", Person.class);

p.chilling();

context.close();

}

}