# 概述

## 简介

。

# 知识点

## Bean Lifecycle

Spring中一个Bean的lifecycle包括如下几个大步骤：

1. 实例化Instantiate
2. 属性注入
3. （可选）获取Spring容器信息；如果实现了xxxAware接口，则可以通过调用对应方法来获取有关Spring容器的信息；
4. （可选）bean预初始化Initialization
5. （可选）Spring初始化bean
6. （可选）自定义初始化
7. （可选）bean后初始化
8. 使用bean
9. （可选）Spring销毁bean
10. （可选）自定义销毁bean

详细的说明可以参考demo\_root中spring\_schema\_extension模块的com.lc.spring.lifecycle.

### 实例化Instantiate

第一步的实例化比较简单，就是调用bean对应的无参构造方法实例化一个对象。其中需要注意的是bean对应的class必须要无参构造方法（不管是java自己生成还是开发时显示声明），如果没有无参构造方法则会报错，无法Instantiate。

### 属性注入

属性注入的形式包括多种，比如setter注入，注解Autowired等。

Setter注入是针对XML文件中bean定义了property标签时执行，对应调用的是setter方法，如果没有定义property标签则不会调用setter。

需要注意的是setter调用顺序和property定义的顺序无关，和class中属性声明顺序有关。

到此一个bean需要通过spring注入的属性都已完成，但是除了这些属性外，一个bean可能还需要其他属性，但是也不需要spring来注入，需要类自身去设置，这就引出后续的初始化Initialization。

### （可选）Aware接口调用

这一步是可选的，因为并不是所有的Bean实例都需要知晓Spring容器相关的信息，如果未实现任何xxxAware接口，这一步将直接跳过。

常见的xxxAware接口有BeanNameAware、ApplicationContextAware等。

### （可选）bean预初始化

这一步的预初始化是只有当系统自定义了BeanPostProcessor的子类并实现了postProcessBeforeInitialization方法时，每一个业务bean（spring系统需要的bean可能不会）在初始化前都会调用这个方法。因此实现这个方法可以做一下初始化前的加工。

### （可选）Spring初始化bean

这一步是当bean实现了InitializingBean接口时，在自定义初始化之前会通过spring来调用接口中的afterPropertiesSet方法。

### （可选）自定义初始化

自定义初始化需要在声明bean定义时通过init-method标签来显示说明需要调用的初始化方法。

### （可选）bean后初始化

当bean初始化完成后，BeanPostProcessor接口中的postProcessAfterInitialization方法会被执行，用于完成初始化后的工作，比如资源的释放之类的。

### 使用bean

### （可选）Spring销毁bean

当Spring Container调用了shutdown方法同时该bean实现类DisposableBean接口时，Spring会调用对应的销毁方法来释放资源。

### （可选）自定义销毁bean

自定义销毁bean需要显示的在bean定义中通过destroy-method标签来说明。