

## Spring注解驱动开发第14讲——你了解@PostConstruct注解和@PreDestroy注解吗?

### 写在前面

在之前的文章中，我们介绍了如何使用@Bean注解指定 初始化 和销毁的方法，也介绍了使用InitializingBean和DisposableBean来处理bean的初始化和销毁。除此之外，在JDK 中还提供了两个注解能够在bean创建完成并且属性赋值完成之后执行一些初始化工作和在容器销毁bean之前通知我们进行一些清理工作。今天，我们就一起来看看这两个注解的用法。

### @PostConstruct注解 和@PreDestroy注解

#### @PostConstruct注解

@PostConstruct注解好多人以为是Spring提供的，其实它是Java自己的注解，是JSR-250规范里面定义的一个注解。我们来看下@PostConstruct注解的源码，如下所示。

```
2+ * Copyright (c) 2005, 2013, Oracle and/or its affiliates. All rights reserved.
25
26 package javax.annotation;
27
28 import java.lang.annotation.*;
29 import static java.lang.annotation.ElementType.*;
30 import static java.lang.annotation.RetentionPolicy.*;
31
33+ * The PostConstruct annotation is used on a method that needs to be executed
77 @Documented
78 @Retention (RUNTIME)
79 @Target(METHOD)
80 public @interface PostConstruct {
81 }
82
```

从源码可以看出，@PostConstruct注解是Java中的注解，并不是Spring提供的注解。

@PostConstruct注解被用来修饰一个非静态的void()方法。被@PostConstruct注解修饰的方法会在服务器加载Servlet的时候运行，并且只会被服务器执行一次。被@PostConstruct注解修饰的方法通常在构造函数之后，init ()方法之前执行。

通常我们是在Spring框架中使用到@PostConstruct注解的，该注解的方法在整个bean初始化中的执行顺序如下：

Constructor（构造方法）→@Autowired（依赖注入）→@PostConstruct（注释的方法）

#### @PreDestroy注解

@PreDestroy注解同样是Java提供的，它也是JSR-250规范里面定义的一个注解。看下它的源码，如下所示。

```
2+ * Copyright (c) 2005, 2013, Oracle and/or its affiliates. All rights reserved.
25
26 package javax.annotation;
27
28 import java.lang.annotation.*;
29 import static java.lang.annotation.ElementType.*;
30 import static java.lang.annotation.RetentionPolicy.*;
31
33+ * The PreDestroy annotation is used on methods as a callback notification to
77 @Documented
78 @Retention (RUNTIME)
79 @Target(METHOD)
80 public @interface PreDestroy {
81 }
82
83
```

被@PreDestroy注解修饰的方法会在服务器卸载Servlet的时候运行，并且只会被服务器调用一次，类似于Servlet的destroy()方法。被@PreDestroy注解修饰的方法会在destroy()方法之后，Servlet被彻底卸载之前执行。执行顺序如下所示：

调用destroy()方法→@PreDestroy→destroy()方法→bean销毁

## 小结

@PostConstruct和@PreDestroy是Java规范JSR-250引入的注解，定义了对对象的创建和销毁工作，同一期规范中还有@Resource注解，Spring也支持了这些注解。

## 一个案例

对@PostConstruct注解和@PreDestroy注解有了简单的了解之后，接下来，我们就写一个简单的程序来加深对这两个注解的理解。

首先，我们创建一个Dog类，如下所示，注意在该类上标注了一个@Component注解。

```
1 package com.meimeixia.bean;
2
3 import javax.annotation.PostConstruct;
4 import javax.annotation.PreDestroy;
5
6 import org.springframework.stereotype.Component;
7
8 /**
9  *
10  * @author liayun
11  *
12  */
13 @Component
14 public class Dog {
15
16     public Dog() {
17         System.out.println("dog constructor...");
18     }
19
20     // 在对象创建完成并且属性赋值完成之后调用
21     @PostConstruct
22     public void init() {
23         System.out.println("dog...@PostConstruct...");
24     }
25
26     // 在容器销毁（移除）对象之前调用
27     @PreDestroy
28     public void destory() {
29         System.out.println("dog...@PreDestroy...");
30     }
31
32 }
```

AI写代码java运行



可以看到，在以上Dog类中，我们提供了构造方法、init()方法以及destory()方法，并且还使用了@PostConstruct注解和@PreDestroy注解来分别标注init()方法和destory()方法。

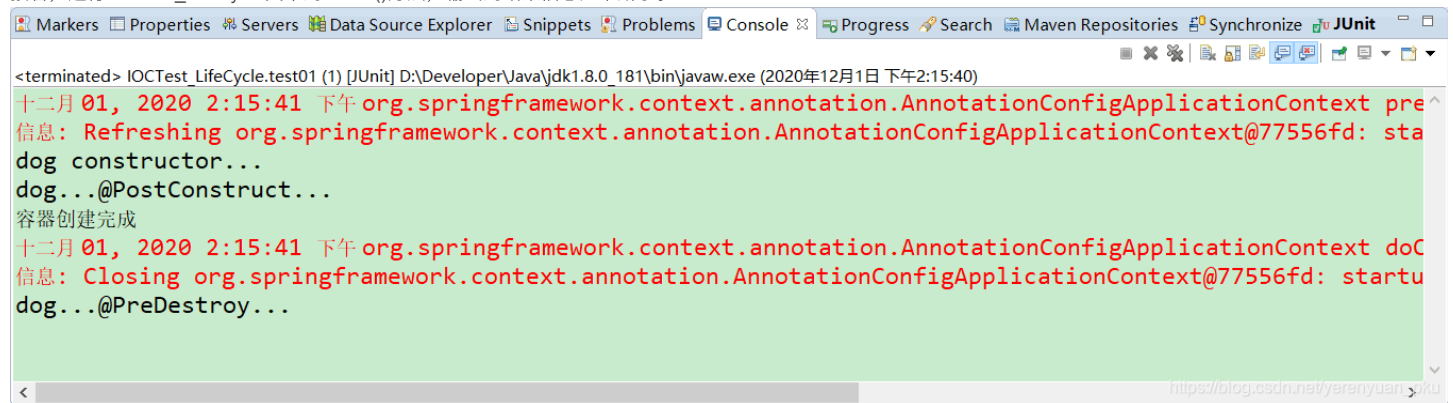
然后，在MainConfigOfLifeCycle配置类中通过包扫描的方式将以上类注入到Spring容器中。

```
1 package com.meimeixia.config;
2
3 import org.springframework.context.annotation.Bean;
4 import org.springframework.context.annotation.ComponentScan;
5 import org.springframework.context.annotation.Configuration;
6 import org.springframework.context.annotation.Scope;
7
8 import com.meimeixia.bean.Car;
9
10 @ComponentScan("com.meimeixia.bean")
11 @Configuration
12 public class MainConfigOfLifeCycle {
13
14     @Scope("prototype")
15     @Bean(initMethod="init", destroyMethod="destroy")
16     public Car car() {
17         return new Car();
18     }
19
20 }
```

AI写代码java运行



接着，运行IOCTest\_LifeCycle类中的test01()方法，输出的结果信息如下所示。



```
<terminated> IOCTest_LifeCycle.test01 (1) [JUnit] D:\Developer\Java\jdk1.8.0_181\bin\javaw.exe (2020年12月1日 下午2:15:40)
十二月 01, 2020 2:15:41 下午 org.springframework.context.annotation.AnnotationConfigApplicationContext pre
信息: Refreshing org.springframework.context.annotation.AnnotationConfigApplicationContext@77556fd: sta
dog constructor...
dog...@PostConstruct...
容器创建完成
十二月 01, 2020 2:15:41 下午 org.springframework.context.annotation.AnnotationConfigApplicationContext doC
信息: Closing org.springframework.context.annotation.AnnotationConfigApplicationContext@77556fd: startu
dog...@PreDestroy...
```

从输出的结果信息中可以看出，被@PostConstruct注解修饰的方法是在bean创建完成并且属性赋值完成之后才执行的，而被@PreDestroy注解修饰的方法是在容器销毁bean之前执行的，通常是进行一些清理工作。