



链滴

# Spring 后置处理器 BeanFactoryPostProcessor 和 BeanPostProcessor 的用法和区别

作者: [jsy](#)

原文链接: <https://ld246.com/article/1493906358911>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

# BeanPostProcessor接口作用是：

如果我们在Spring容器完成Bean的实例化、配置和其他的初始化前后添加一些自己的逻辑处理，我们就可以定义一个或者多个BeanPostProcessor接口的实现，然后注册到容器中

BeanPostProcessor接口有两个方法需要实现：postProcessBeforeInitialization和postProcessAfterInitialization，前者在实例化及依赖注入完成后、在任何初始化代码（比如配置文件中的init-method）调用之前调用；后者在初始化代码调用之后调用。

注意：

1、接口中的两个方法都要将传入的bean返回，而不能返回null，如果返回的是null那么我们通过getBean方法将得不到目标。

```
public class StepHandlerBeanPostProcessor implements BeanPostProcessor {  
    private StepWorkerHolder stepWorkerHolder;  
  
    public StepHandlerBeanPostProcessor(StepWorkerHolder stepWorkerHolder) {  
        this.stepWorkerHolder = stepWorkerHolder;  
    }  
  
    @Override  
    public Object postProcessBeforeInitialization(Object bean, String beanName)  
            throws BeansException {  
        return bean;  
    }  
  
    @Override  
    public Object postProcessAfterInitialization(Object bean, String beanName)  
            throws BeansException {  
        processStepHandlerAnnotation(bean);  
        return bean;  
    }  
  
    private void processStepHandlerAnnotation(Object bean) {  
        final Class beanClass = AopProxyUtils.ultimateTargetClass(bean);  
        StepHandler stepHandlerAnnotation = beanClass.getAnnotation(StepHandler.class);  
        if (stepHandlerAnnotation == null) {  
            return;  
        }  
  
        // 记录stepWorker，这样就可以把包含注解StepHandler的bean都存到缓存里面  
        stepWorkerHolder.putStepWorker(stepHandlerAnnotation.value(), (StepWorker) bean)  
    }  
}  
  
@Target(ElementType.TYPE)  
@Retention(RetentionPolicy.RUNTIME)  
public @interface StepHandler {  
    String value();  
}
```

## BeanFactoryPostProcessor可以修改BEAN的配置信息而BeanPostProcessor不能

BeanFactoryPostProcessor的回调比BeanPostProcessor要早，还有就是BeanFactoryPostProcessor确实有能力改变初始化BEAN的内容

```
package com.springdemo.postProcessor;

import org.springframework.beans.BeansException;
import org.springframework.beans.MutablePropertyValues;
import org.springframework.beans.factory.config.BeanDefinition;
import org.springframework.beans.factory.config.BeanFactoryPostProcessor;
import org.springframework.beans.factory.config.ConfigurableListableBeanFactory;

public class MyBeanFactoryPostProcessor implements BeanFactoryPostProcessor {

    public void postProcessBeanFactory(ConfigurableListableBeanFactory beanFactory)
        throws BeansException {
        // TODO Auto-generated method stub
        //BeanFactoryPostProcessor可以修改BEAN的配置信息而BeanPostProcessor不能
        //我们在这里修改postProcessorBean的username注入属性
        BeanDefinition bd = beanFactory.getBeanDefinition("postProcessorBean");
        MutablePropertyValues pv = bd.getPropertyValues();
        if(pv.contains("username")){
            {
                pv.addPropertyValue("username", "xiaojun");
            }
        }
    }
}
```