Spring JSR-250 注解

## **Spring JSR-250 注解**

Spring还使用基于 JSR-250 注解，它包括 @PostConstruct， @PreDestroy 和 @Resource 注解。因为你已经有了其他的选择，尽管这些注解并不是真正所需要的，但是关于它们仍然让我给出一个简短的介绍。

## **@PostConstruct 和 @PreDestroy 注解：**

为了定义一个 bean 的安装和卸载，我们使用 ****init-method**** 和/或 ****destroy-method**** 参数简单的声明一下 。init-method 属性指定了一个方法，该方法在 bean 的实例化阶段会立即被调用。同样地，destroy-method 指定了一个方法，该方法只在一个 bean 从容器中删除之前被调用。

你可以使用 ****@PostConstruct**** 注解作为初始化回调函数的一个替代，****@PreDestroy**** 注解作为销毁回调函数的一个替代，其解释如下示例所示。

## **示例**

这里是 ****HelloWorld.java**** 文件的内容：

package com.tutorialspoint;import javax.annotation.\*;public class HelloWorld {

private String message;

public void setMessage(String message){

this.message = message;

}

public String getMessage(){

System.out.println("Your Message : " + message);

return message;

}

@PostConstruct

public void init(){

System.out.println("Bean is going through init.");

}

@PreDestroy

public void destroy(){

System.out.println("Bean will destroy now.");

}

}

下面是 ****MainApp.java**** 文件的内容。这里你需要注册一个关闭钩 ****registerShutdownHook()**** 方法，该方法在 AbstractApplicationContext 类中被声明。这将确保一个完美的关闭并调用相关的销毁方法。

package com.tutorialspoint;import org.springframework.context.support.AbstractApplicationContext;import org.springframework.context.support.ClassPathXmlApplicationContext;public class MainApp {

public static void main(String[] args) {

AbstractApplicationContext context =

new ClassPathXmlApplicationContext("Beans.xml");

HelloWorld obj = (HelloWorld) context.getBean("helloWorld");

obj.getMessage();

context.registerShutdownHook();

}

}

下面是配置文件 ****Beans.xml****，该文件在初始化和销毁方法中需要使用。

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

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

xmlns:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-3.0.xsd

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context-3.0.xsd">

<context:annotation-config/>

<bean id="helloWorld"

class="com.tutorialspoint.HelloWorld"

init-method="init" destroy-method="destroy">

<property name="message" value="Hello World!"/>

</bean>

</beans>

一旦你在源文件和 bean 配置文件中完成了上面两处改变，让我们运行一下应用程序。如果你的应用程序一切都正常的话，这将会输出以下消息：

Bean is going through init.

Your Message : Hello World!

Bean will destroy now.

## **@Resource 注解：**

你可以在字段中或者 setter 方法中使用 ****@Resource**** 注解，它和在 Java EE 5 中的运作是一样的。@Resource 注解使用一个 ‘name’ 属性，该属性以一个 bean 名称的形式被注入。你可以说，它遵循 ****by-name**** 自动连接语义，如下面的示例所示：

package com.tutorialspoint;import javax.annotation.Resource;public class TextEditor {

private SpellChecker spellChecker;

@Resource(name= "spellChecker")

public void setSpellChecker( SpellChecker spellChecker ){

this.spellChecker = spellChecker;

}

public SpellChecker getSpellChecker(){

return spellChecker;

}

public void spellCheck(){

spellChecker.checkSpelling();

}

}

如果没有明确地指定一个 ‘name’，默认名称源于字段名或者 setter 方法。在字段的情况下，它使用的是字段名；在一个 setter 方法情况下，它使用的是 bean 属性名称。