

05 Spring_配置详解_Bean元素进阶

Spring

一：实体类

```
1. public class Users {
2.     private Integer id;
3.     private String name;
4.
5.     public Users() {
6.         System.out.println("创建了一个对象");
7.     }
8.     //对应的get set 方法
9. }
```

二：scope属性

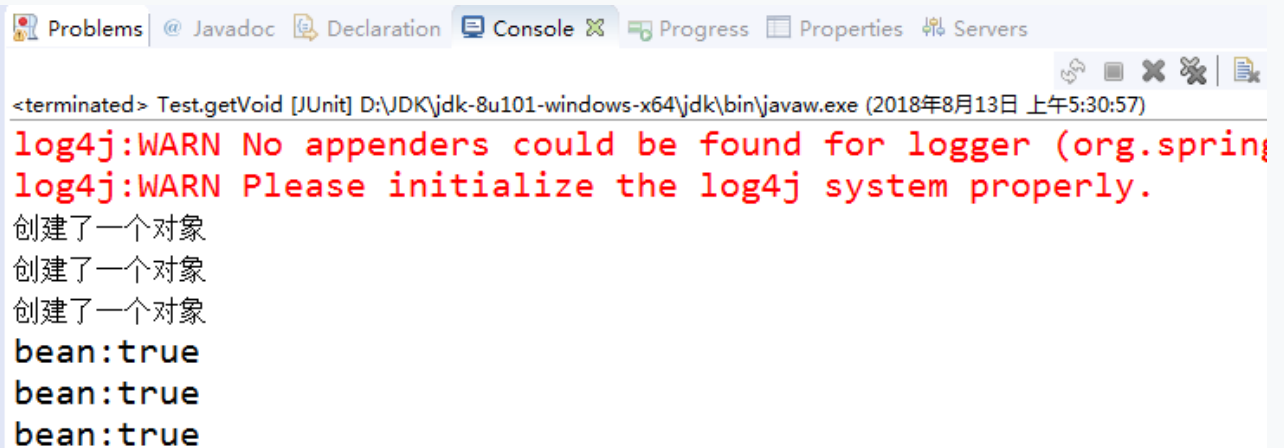
```
1. <!--
2.     scope属性(常用):
3.         |-singleton(默认值):单列对象,在Spring容器中只会有一个对象
4.         |-prototype:多列原型,在Spring容器中每次都会方式新的对象
5.         |-session(了解):web环境中,对象与session生命周期一致
6.         |-request(了解):web环境中,对象与request生命周期一致
7.         总结:在一般情况下使用默认值
8.         注意在Strtus和Spring整合是Strtus的actionBean必须为配置多
9.         列的
10.     -->
11.     <bean name="users" class="com.spring.bean.Users"
12.         scope=""
13.     >
14. </bean>
```

1、scope="singleton"("默认")

```

1.      @org.junit.Test
2.      public void getVoid(){
3.          //1.创建容器对象
4.          ClassPathXmlApplicationContext classPath= new
ClassPathXmlApplicationContext("applicationContext.xml");
5.          //2.找容器要对象
6.          Users bean = (Users)classPath.getBean("users");
7.          Users bean1 = (Users)classPath.getBean("users");
8.          Users bean2 = (Users)classPath.getBean("users");
9.          //3.打印对象单列为true
10.         System.out.println("bean:"+ (bean==bean1));
11.         System.out.println("bean:"+ (bean1==bean2));
12.         System.out.println("bean:"+ (bean2==bean));
13.     }

```



The screenshot shows an IDE console window with the following content:

```

<terminated> Test.getVoid [JUnit] D:\JDK\jdk-8u101-windows-x64\jdk\bin\javaw.exe (2018年8月13日 上午5:30:57)
log4j:WARN No appenders could be found for logger (org.springframework.test.util.ReflectionTestUtils).
log4j:WARN Please initialize the log4j system properly.
创建了一个对象
创建了一个对象
创建了一个对象
bean:true
bean:true
bean:true

```

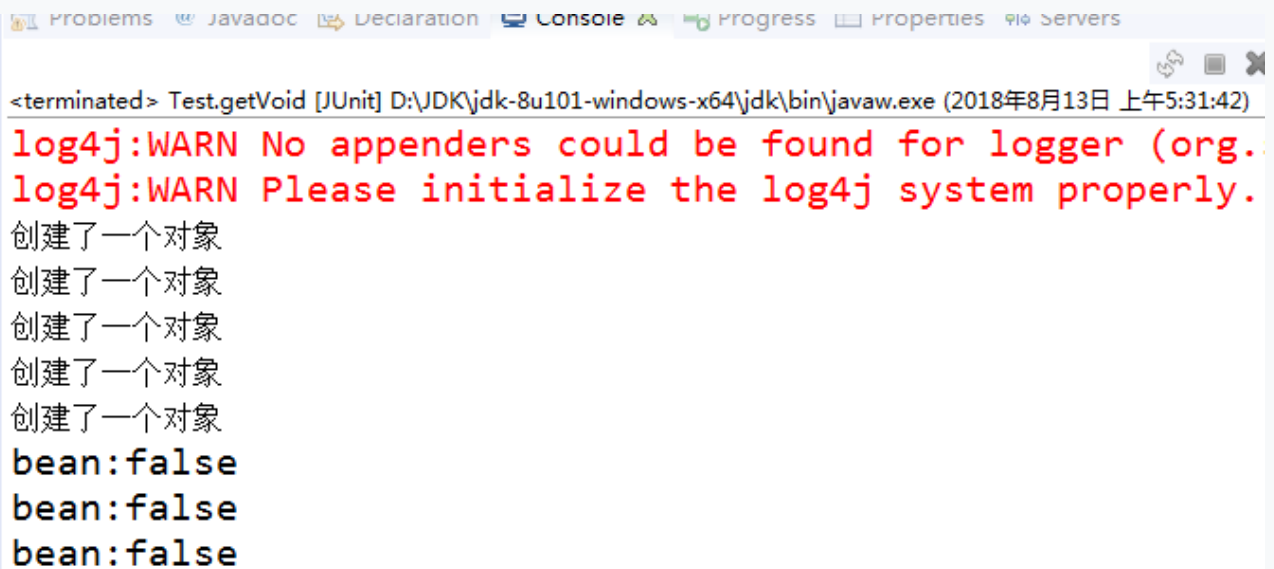
2、scope="prototype"

```

1.      @org.junit.Test
2.      public void getVoid(){
3.          //1.创建容器对象
4.          ClassPathXmlApplicationContext classPath= new
ClassPathXmlApplicationContext("applicationContext.xml");
5.          //2.找容器要对象
6.          Users bean = (Users)classPath.getBean("users");
7.          Users bean1 = (Users)classPath.getBean("users");
8.          Users bean2 = (Users)classPath.getBean("users");
9.          //3.打印对象多列为False
10.         System.out.println("bean:"+ (bean==bean1));
11.         System.out.println("bean:"+ (bean1==bean2));

```

```
12.         System.out.println("bean:"+(bean2==bean));
13.     }
```



The screenshot shows an IDE console window with the following content:

```
<terminated> Test.getVoid [JUnit] D:\JDK\jdk-8u101-windows-x64\jdk\bin\javaw.exe (2018年8月13日 上午5:31:42)
log4j:WARN No appenders could be found for logger (org.
log4j:WARN Please initialize the log4j system properly.
创建了一个对象
创建了一个对象
创建了一个对象
创建了一个对象
创建了一个对象
bean:false
bean:false
bean:false
```

三：生命周期属性

- 1、生命周期初始化方法，spring在这个对象创建之后调用
- 2、生命周期销毁的方法，spring容器在关闭并销毁所有容器中的对象

```
1.     <!--
2.     init-method:初始化 实体类中必须有方法
3.     destroy-method:销毁 实体类中必须有方法
4.     -->
5.     <bean    name="users" class="com.spring.bean.Users"
6.         init-method="init" destroy-method="destroy"
7.     >
8. </bean>
```

1、实体类中必须有方法

```
1.     public void init(){
2.         System.out.println("init初始化的方法");
3.     }
```

```

4.
5.     public void destroy(){
6.         System.out.println("destroy初始化的方法");
7.     }

```

2、测试

```

1.     @org.junit.Test
2.     public void getVoid01(){
3.         //1.创建容器对象
4.         ClassPathXmlApplicationContext classPath= new
ClassPathXmlApplicationContext("applicationContext.xml");
5.         //2.找容器要对象
6.         Users bean = (Users)classPath.getBean("users");
7.         //3.打印对象单列为true
8.         System.out.println("bean:"+bean);
9.         classPath.close();
10.    }

```

```

log4j:WARN No appenders could be found for logger
log4j:WARN Please initialize the log4j system prop

```

创建了一个对象

init初始化的方法

创建了一个对象

创建了一个对象

bean:com.spring.bean.Users@1b26f7b2

destroy初始化的方法

四：模块化配置_资源路径

1、导入配置

1、在一个配置文件中导入其他的配置文件

```
1. <beans xmlns="http://www.springframework.org/schema/beans"
2.     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3.     xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">
4.     <!--相对的路径地址-->
5.     <import resource="com/spring/bean/applicationContext.xml"/>
6. </beans>
```

2、使用Ant-style风格的路径

1、三个通配符

|-? 匹配任何单字符

|-* 匹配0或者任意数量的字符

|-** 匹配0或者更多的目录