链滴

# Spring 系列之注解开启 annotation-config 和注解扫描 component-scan 区分

作者：wubo8196

原文链接：https://ld246.com/article/1575367780401

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)

# 前言

Spring框架对Bean进行装配提供了很灵活的方式，下面归纳一下主要的方式：

- 在XML中进行显示配置
- 在Java中进行显示配置
- 隐式的bean发现机制和自动装配

而自动装配就需要注解扫描，这里有两种开启注解扫描的方式，即

<context:annotation-config/>

和

<context:component-scan>

下面归纳一下这两种注解方式的异同点：

<context:annotation-config/>**注解扫描是针对已经在Spring容器里注册过的Bean**

<context:component-scan/>不仅具备<context:annotation-config/>的所有功能，还可以在指定package下面扫描对应的bean

---

## XML注册Bean方式

下面给出两个类，类A和类B

```
package com.test;
```

```
public class B{
    public B(){
        System.out.println("B类");
    }
}

package com.test;

public class A{
    private B b;
    public void setB(B b){
        this.b = b;
        System.out.println("通过set的方式注入B类");
    }

    public A(){
        System.out.println("A类");
    }
}
```

然后在Spring的Application.xml里作如下配置，不开启注解扫描

```
    <bean id="bBean" class="com.test.B"/>
    <bean id="aBean" class="com.test.A">
        <property name="b" ref="bBean"/>
    </bean>
```

启动加载Application.xml 得到如下输出

```
B类
A类
通过set的方式注入B类
```

分析：

实例化B类，实例化A类并注入B的实例

---

## annotation配置注解开启的方式

同样两个类，类A和类B，此处在A类中通过@Autowired的方式注入B的实例

```
package com.test;
public class B{
    public B(){
        System.out.println("B类");
    }
}

package com.test;
public class A{
    private B b;
    public A(){
```

```
        System.out.println("A类");
    }

    @Autowired
    public void setB(B b){
        this.b = b; System.out.println("通过set的方式注入B类");
    }
}
```

然后做如下测试

1.仅仅在Application.xml里注册Bean，不开启扫描

```
    <bean id="bBean"class="com.test.B"/>
        <bean id="aBean"class="com.test.A"/>
```

2.仅仅开启扫描，不注册Bean

```
    <context:annocation-config/>
```

以上两种配置方式配置启动加载Application.xml，得到如下一致结果

```
B类
A类
```

会发现@Autowired下的方法没有执行，即没有自动注入

然后修改Application.xml，开启注解

```
    <context:annocation-config/>
    <bean id="bBean"class="com.test.B"/>
        <bean id="aBean"class="com.test.A"/>
```

重新启动加载Application.xml，得到如下输出

```
B类
A类
通过set的方式注入B类
```

归纳：<context:annotation-config>:**注解扫描是针对已经在Spring容器里注册过的Bean**

---

## component配置注解开启方式

同样两个类，类A和类B，此处在B类添加@Component注解

```
package com.test;
public class B{
    public B(){
        System.out.println("B类");
    }
}

package com.test;
@Component
```

```
public class A{
    private B b;
    public A(){
        System.out.println("A类");
    }

    @Autowired
    public void setB(B b){
        this.b = b; System.out.println("通过set的方式注入B类");
    }
}
```

然后配置一下application.xml，开启annotaion-config扫描

```
    <context:annocation-config/>
```

重新启动加载Application.xml，得到如下输出

```
B类
A类
```

原因：<context:annotation-config>:**注解扫描是针对已经在Spring容器里注册过的Bean**，Bean并有注册过，所以即使开启了@Autowired、@Component注解 和配置开启了annotaion-config扫描是加载不到

修改配置文件：

```
<context:component-scan base-package="com.test"/>
```

重新启动加载Application.xml，得到如下输出

```
B类
A类
通过set的方式注入B类
```

总结：

<context:annotation-config>:**注解扫描是针对已经在Spring容器里注册过的Bean**

<context:component-scan>:不仅具备<context:annotation-config>的所有功能，还可以在指定的ackage下面扫描对应的bean

<context:annotation-config />和 <context:component-scan>同时存在的时候，前者会被忽略。

即使注册Bean，同时开启<context:annotation-config />扫描，@autowire，@resource等注入解只会被注入一次，也即只加载一次