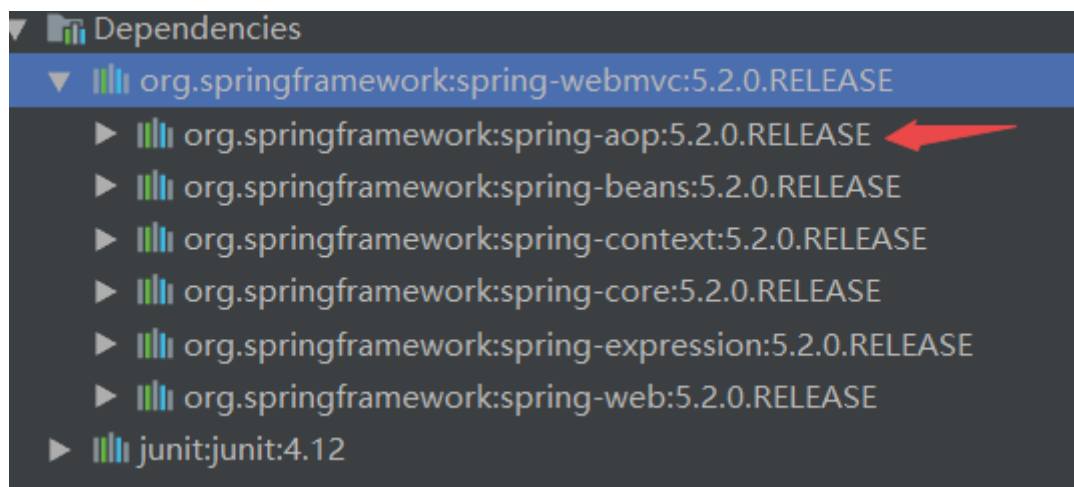


使用注解开发

- 在Spring4之后，使用注解开发，必须要保证aop的包导入了



- 使用注解必须添加context的约束，开启注解支持

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:aop="http://www.springframework.org/schema/aop"
6       xsi:schemaLocation="http://www.springframework.org/schema/beans
7                           https://www.springframework.org/schema/beans/spring-
8                           beans.xsd
9                           http://www.springframework.org/schema/context
10                          https://www.springframework.org/schema/context/spring-
11                          context.xsd
12                          http://www.springframework.org/schema/aop
13                          https://www.springframework.org/schema/aop/spring-aop.xsd">
14     <!--开启注解的支持-->
15     <context:annotation-config/>
16 </beans>
```

1 @Component —— 注入bean

组件，放在类上说明了这个类被Spring管理了，就是bean。

```
1 //等价于 <bean id="user" class="com.kuang.pojo.User"/>
2 //Component 组件
3 @Component
4 public class User implements Serializable {
5     public String name = "Geekst";
6 }
```

```

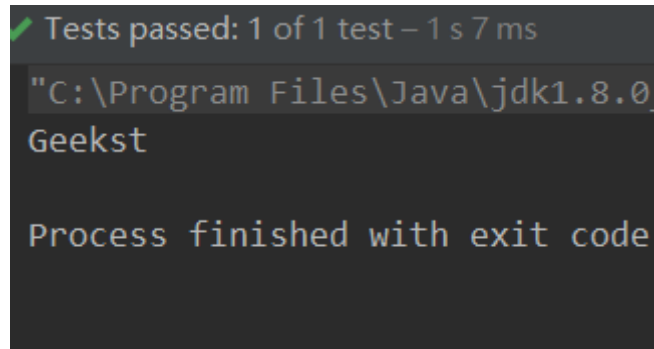
1 <!--指定要扫描的包，这个包下的注解就会生效-->
2 <context:component-scan base-package="com.kuang.pojo"/>
3
4 <!--开启注解的支持-->
5 <context:annotation-config/>

```

```

1 @Test
2 public void test(){
3     ApplicationContext context = new
    ClassPathXmlApplicationContext("applicationContext.xml");
4     User user = context.getBean("user", User.class); //名字默认为 类首字母小写
5
6     System.out.println(user.name);
7 }

```



```

✓ Tests passed: 1 of 1 test - 1 s 7 ms
"C:\Program Files\Java\jdk1.8.0
Geekst
Process finished with exit code

```

2 @Value —— 注入属性

```

1 //等价于 <property name="name" value="GeekByValue"/>
2 @Value("GeekByValue")
3 public String name;

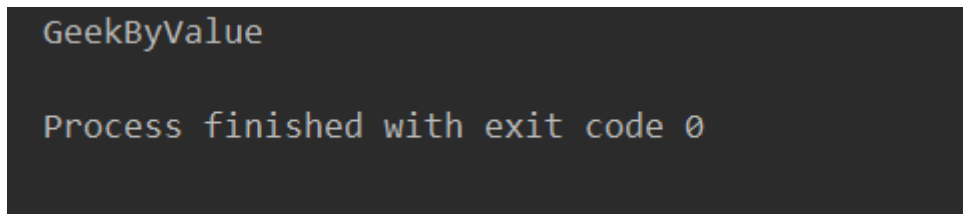
```

也可以在set方法上注入：

```

1 @Value("GeekByValue")
2 public void setName(String name) {
3     this.name = name;
4 }

```



```

GeekByValue
Process finished with exit code 0

```

3 衍生的注解

由@Component衍生的：

- dao层 —— @Repository 【一般用不到】
- service层 —— @Service
- Controller层 —— @Controller

虽然注解名字不一样，但是功能都相同，都是注册到容器中。

但是，记得加上扫描所有的包：

```
1 | <context:component-scan base-package="com.kuang"/>
```

4 自动装配

详见 07 自动装配bean

@Autowired

@Resource

5 作用域

```
1 | @Scope("singleton") //标注为单例模式
```

6 小结

- xml更加万能，适用于任何场合，维护简单方便
- 注解：不是自己的类使用不了，维护相对复杂
- 最佳实践：
 - xml用来管理bean（不用那些Component和衍生的注解）
 - 注解只负责完成属性的注入
 - 在使用的规程中，只需要注意一个问题：想让注解生效，必须要开启注解的支持

```
1 | <!--指定要扫描的包，这个包下的注解就会生效-->
2 | <context:component-scan base-package="com.kuang"/>
3 |
4 | <!--开启注解的支持-->
5 | <context:annotation-config/>
```