[Spring@Autowired注解与自动装配](http://blog.csdn.net/heyutao007/article/details/5981555)

分类： [Web](http://blog.csdn.net/HEYUTAO007/article/category/680606)2010-11-02 11:40 136421人阅读 [评论](http://blog.csdn.net/heyutao007/article/details/5981555#comments)(29) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/heyutao007/article/details/5981555#report)

[autowired](http://www.csdn.net/tag/autowired)[spring](http://www.csdn.net/tag/spring)[bean](http://www.csdn.net/tag/bean)[office](http://www.csdn.net/tag/office)[class](http://www.csdn.net/tag/class)[encoding](http://www.csdn.net/tag/encoding)

**1   配置文件的方法**

我们编写spring 框架的代码时候。一直遵循是这样一个规则：所有在spring中注入的bean 都建议定义成私有的域变量。并且要配套写上 get 和 set方法。

|  |
| --- |
| Boss 拥有 Office 和 Car 类型的两个属性：          清单 3. Boss.java    **[java]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)   1. **package** com.baobaotao; 3. **public** **class** Boss { 4. **private** Car car; 5. **private** Office office; 7. // 省略 get/setter 9. @Override 10. **public** String toString() { 11. **return** "car:" + car + "/n" + "office:" + office; 12. } 13. }          **System.out.println必须实现toString方法**    我们在 Spring 容器中将 Office 和 Car 声明为 Bean，并注入到 Boss Bean 中：下面是使用传统 XML 完成这个工作的配置文件 beans.xml：          清单 4. beans.xml 将以上三个类配置成 Bean     **[xhtml]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)   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. xsi:schemaLocation="http://www.springframework.org/schema/beans 5. http://www.springframework.org/schema/beans/spring-beans-2.5.xsd"**>** 6. **<bean** id="boss" class="com.baobaotao.Boss"**>** 7. **<property** name="car" ref="car"**/>** 8. **<property** name="office" ref="office" **/>** 9. **</bean>** 10. **<bean** id="office" class="com.baobaotao.Office"**>** 11. **<property** name="officeNo" value="002"**/>** 12. **</bean>** 13. **<bean** id="car" class="com.baobaotao.Car" scope="singleton"**>** 14. **<property** name="brand" value=" 红旗 CA72"**/>** 15. **<property** name="price" value="2000"**/>** 16. **</bean>** 17. **</beans>**              当我们运行以下代码时，控制台将正确打出 boss 的信息：          清单 5. 测试类：AnnoIoCTest.java     **[java]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)   1. **import** org.springframework.context.ApplicationContext; 2. **import** org.springframework.context.support.ClassPathXmlApplicationContext; 3. **public** **class** AnnoIoCTest { 5. **public** **static** **void** main(String[] args) { 6. String[] locations = {"beans.xml"}; 7. ApplicationContext ctx = 8. **new** ClassPathXmlApplicationContext(locations); 9. Boss boss = (Boss) ctx.getBean("boss"); 10. System.out.println(boss); 11. } 12. }         这说明 Spring 容器已经正确完成了 Bean 创建和装配的工作。 |

**2   @Autowired**

Spring 2.5 引入了 @Autowired 注释，它可以对类成员变量、方法及构造函数进行标注，完成自动装配的工作。 通过 @Autowired的使用来消除 set ，get方法。

要实现我们要精简程序的目的。需要这样来处理：   
  
**\* 在applicationContext.xml中加入：** 

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)

1. <!-- 该 BeanPostProcessor 将自动对标注 @Autowired 的 Bean 进行注入 -->
2. <bean **class**="org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor"/>

Spring 通过一个 BeanPostProcessor 对 @Autowired 进行解析，所以要让 @Autowired 起作用必须事先在 Spring 容器中声明 AutowiredAnnotationBeanPostProcessor Bean。

**\* 修改在原来注入spirng容器中的bean的方法。**   
     在域变量上加上标签@Autowired,并且去掉 相应的get 和set方法

清单 6. 使用 @Autowired 注释的 Boss.java   

**[java]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)

1. **package** com.baobaotao;
2. **import** org.springframework.beans.factory.annotation.Autowired;
4. **public** **class** Boss {
6. @Autowired
7. **private** Car car;
9. @Autowired
10. **private** Office office;
12. …
13. }

**\* 在applicatonContext.xml中 把原来 引用的<porpery >标签也去掉。**

**[xhtml]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)

2. **<?xml** version="1.0" encoding="UTF-8" **?>**
3. **<beans** xmlns="http://www.springframework.org/schema/beans"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xsi:schemaLocation="http://www.springframework.org/schema/beans
6. http://www.springframework.org/schema/beans/spring-beans-2.5.xsd"**>**
8. <!-- 该 BeanPostProcessor 将自动起作用，对标注 @Autowired 的 Bean 进行自动注入 -->
9. **<bean** class="org.springframework.beans.factory.annotation.
10. AutowiredAnnotationBeanPostProcessor"**/>**
12. <!-- 移除 boss Bean 的属性注入配置的信息 -->
13. **<bean** id="boss" class="com.baobaotao.Boss"**/>**
15. **<bean** id="office" class="com.baobaotao.Office"**>**
16. **<property** name="officeNo" value="001"**/>**
17. **</bean>**
18. **<bean** id="car" class="com.baobaotao.Car" scope="singleton"**>**
19. **<property** name="brand" value=" 红旗 CA72"**/>**
20. **<property** name="price" value="2000"**/>**
21. **</bean>**
22. **</beans>**

 这样，当 Spring 容器启动时，AutowiredAnnotationBeanPostProcessor 将扫描 Spring 容器中所有 Bean，当发现 Bean 中拥有 @Autowired 注释时就找到和其匹配（默认按类型匹配）的 Bean，并注入到对应的地方中去。     
    
按照上面的配置，Spring 将直接采用 Java 反射机制对 Boss 中的 car 和 office 这两个私有成员变量进行自动注入。所以对成员变量使用 @Autowired 后，您大可将它们的 setter 方法（setCar() 和 setOffice()）从 Boss 中删除。     
    
当然，您也可以通过 @Autowired 对方法或构造函数进行标注，如果构造函数有两个入参，分别是 bean1 和 bean2，@Autowired 将分别寻找和它们类型匹配的 Bean，将它们作为 CountryService (Bean1 bean1 ,Bean2 bean2) 的入参来创建 CountryService Bean。来看下面的代码：  对方法

**[java]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)

1. **package** com.baobaotao;
3. **public** **class** Boss {
4. **private** Car car;
5. **private** Office office;
7. @Autowired
8. **public** **void** setCar(Car car) {
9. **this**.car = car;
10. }
12. @Autowired
13. **public** **void** setOffice(Office office) {
14. **this**.office = office;
15. }
16. …
17. }

这时，@Autowired 将查找被标注的方法的入参类型的 Bean，并调用方法自动注入这些 Bean。而下面的使用方法则对构造函数进行标注：

**[java]** [view plaincopyprint?](http://blog.csdn.net/heyutao007/article/details/5981555)

1. **package** com.baobaotao;
3. **public** **class** Boss {
4. **private** Car car;
5. **private** Office office;
7. @Autowired
8. **public** Boss(Car car ,Office office){
9. **this**.car = car;
10. **this**.office = office ;
11. }
13. …
14. }

由于 Boss() 构造函数有两个入参，分别是 car 和 office，@Autowired 将分别寻找和它们类型匹配的 Bean，将它们作为 Boss(Car car ,Office office) 的入参来创建 Boss Bean。