@Resource注解学习

## Spring的依赖注入几种方式

### 使用**构造器**注入；

### 使用属性**setter方法**注入。

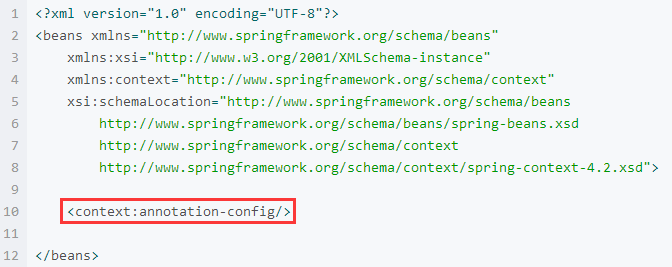
但其实还有一种注入方式，那就是**使用Field注入**（用于注解方式）。

**注入依赖对象**可以采用手工装配或自动装配，在实际应用中建议使用手工装配，因为自动装配会产生未知情况，开发人员无法预见最终的装配结果。 手工装配依赖对象，在这种方式中又有两种编程方式：

### 在xml配置文件中，通过在bean节点下配置，如：



### 在java代码中使用**@Autowired**或**@Resource**注解方式进行装配。但我们需要在xml配置文件中配置以下信息：



利用注解方式注入依赖对象，必须在配置文件中配置注解解析处理的处理器，添加

**<context:annotation-config**/> 。



**这个配置隐式注册了多个对注解进行解析处理的处理器**：

AutowiredAnnotationBeanPostProcessor

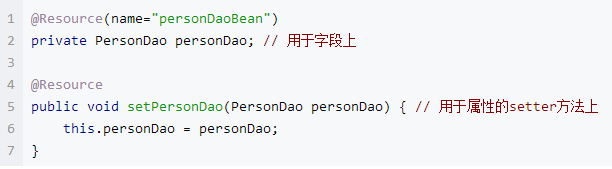
CommonAnnotationBeanPostProcessor

PersistenceAnnotationBeanPostProcessor

RequiredAnnotationBeanPostProcessor

在本文中我们将关注点着重放在@Resource注解上。

@Resource注解和@Autowired一样，同样也可以标注在字段或属性的setter方法上，但它默认按名称装配。名称可以通过@Resource的name属性指定，如果没有指定name属性，当注解标注在字段上时，即默认取字段的名称作为bean名称寻找依赖对象，当注解标注在属性的setter方法上时，即默认取属性名作为bean名称寻找依赖对象。如：



注意：如果没有指定name属性，并且按照默认的名称仍然找不到依赖对象时， @Resource注解会回退到按类型装配。但一旦指定了name属性，就只能按名称装配了。

注意：当我们使用注解方式注入属性时，一定要往项目中导入spring-aop-4.2.5.RELEASE.jar包，不然就会报异常。

# @Autowired与@Resource的区别

在java代码中使用**@Autowired或@Resource**注解方式进行装配。

## 相同点

### **@Autowired**与**@Resource**都可以用来装配bean；都可以写在**字段**上,或写在**setter方法**上。

## 不同点

### @Autowired默认按**类型(type)**装配，@Resource默认按**名称**(name)装配，当找不到与名称匹配的bean才会按类型装配；

### @Autowired属于Spring的，@Resource属于J2EE的；(建议使用@Resource)；

### @Autowired默认按**类型**装配（这个注解是属业spring的），默认情况下必须要求依赖对象必须存在，如果要允许null 值，可以设置它的required属性为false，如：@Autowired(required=false) ，如果我们想使用名称装配可以结合@Qualifier注解进行使用，如下：

Java代码：

@Autowired() @Qualifier("baseDao")

private BaseDao baseDao;

### @Resource（这个注解属于J2EE的），默认安照名称进行装配，名称可以通过name属性进行指定， 如果没有指定name属性，当注解写在字段上时，默认取**字段名**进行按照名称查找，如果注解写在setter方法上默认取属性名进行装配。 当找不到与名称匹配的bean时才按照**类型**进行装配。但是需要注意的是，如果name属性一旦指定，就只会按照名称进行装配。

Java代码：

@Resource(name="baseDao")

private BaseDao baseDao;

推荐使用：**@Resource注解**在字段上，且这个注解是属于**J2EE**的，减少了与spring的耦合。最重要的这样代码看起就比较优雅。