Xml方式省略

# 自动注入

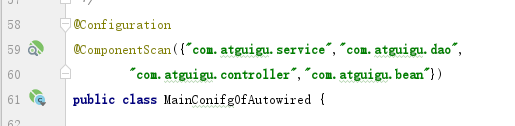
## 自动装配

*自动装配;  
 Spring利用依赖注入（DI），完成对IOC容器中中各个组件的依赖关系赋值；*

*AutowiredAnnotationBeanPostProcessor:解析完成自动装配功能；*

## 配置扫描

Xml方式省略



## 区别

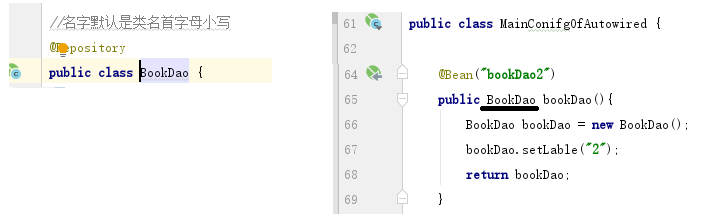
***@Autowired:Spring定义的；*** *@Resource、@Inject都是java规范*

# @Autowired

## 默认

### 作用

*1）、默认优先按照类型去容器中找对应的组:applicationContext.getBean(BookDao.class);找到就赋值*

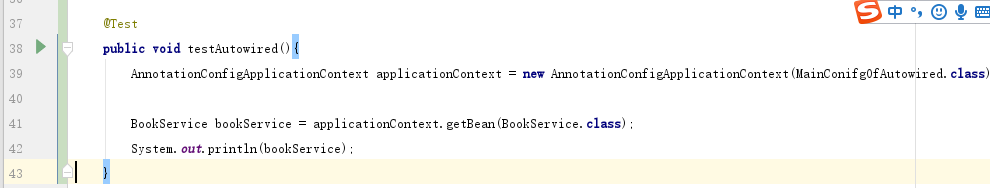


测试spring中找到两个类型BookDao

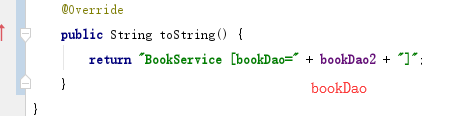
*2）、如果找到多个相同类型的组件，再将属性的名称作为组件的id去容器中查找  
 applicationContext.getBean("bookDao")*

### 测试

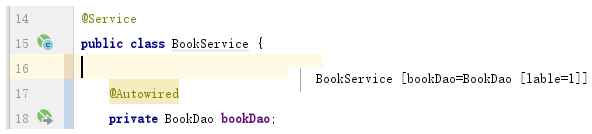
测试：

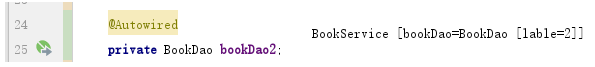


调用toString方法



Service 就看实例名了



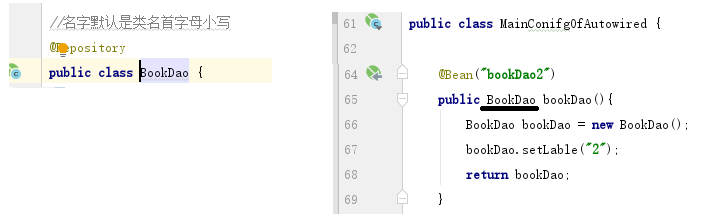


## @Qualifier

### 作用

配合@autowired使用，类型注入不再生效，强制属性注入

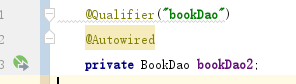
*@Qualifier("bookDao")：使用@Qualifier指定需要装配的组件的id，而不是使用属性名*

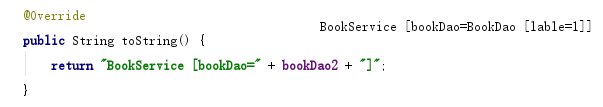


### 测试



就算在这里属性名用的bookDao2，只要上面指定了，容器里就是bookDao

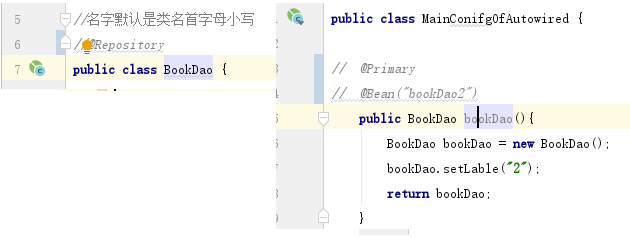




## required=false

### 作用

*自动装配默认一定要将属性赋值好，没有就会报错；  
 可以使用@Autowired(required=false);*

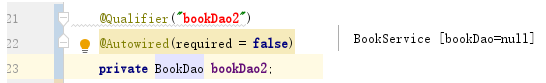


测试spring中没有实例

### 测试



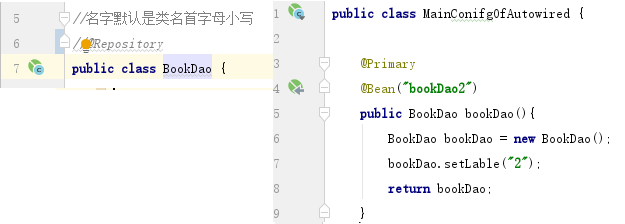
得到结果为空



## @Primary（作用@bean）

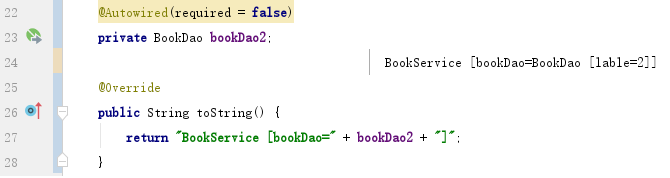
### 作用

*@Primary：让Spring进行自动装配的时候，默认使用首选的bean；  
 也可以继续使用@Qualifier指定需要装配的bean的名字*



### 测试

没有明确属性指定时



当@Qualifier明确指定时

