# 基于注解的配置

**1. @Component 注解**

**此注解用于class的定义之前，如果在xml配置文件中，被：**

<context:component-scan *base-package*="package1.package2…" />

**包含，则这个注解的class资源将被spring容器自动配置。**

**注：可以通过@Component(“beanId”)来指明这个注解对应的bean的id属性值**

**2. @AutoWire 注解**

**此注解可用于数据域或者方法之前，表示自动配置数据域的值或者自动配置方法的形参. 注意：需要自动配置的类型不能是基本类型数据。**

**其原理是直接在spring容器中根据数据类型查找配置好了的资源，即不管该资源如何配置（通过bean或者通过@Component注解都一样），只有保证找到才不会报错，优先匹配通过@Component配置的资源，找到之后将该资源的引用赋值给@AutoWire注解的方法或者变量。**

**注：可以和@Qualifier注解一起使用，通过@Qualifier注解的参数选择特定的id属性对应的bean**

**3. @Required 注解**

**注：下面指的爆红是不会影响程序正常执行的错误**

**这个注解是为了提示作用，即如果没有按照期望要求配置已经注解的资源，则就会在Intellij IDEA开发工具中爆红：**

**如：AClass类中有方法setXxx()被@Required注解，则，在Spring容器中的所有xml文件中，只要存在一个针对于类AClass配置的bean，这个bean中有property子元素或者这个并设置了自动配置属性：autowire，则java中的setXxx方法就不会爆红。否则就会在setXxx这个方法名上爆红提示。**

**而xml文件会因为这个注解在某种情况下也会爆红，这个情况是，对于任何针对类AClass配置的bean，该bean**

**只要不能导致被@Required注解的setXxx方法被调用，则这个bean就会在class属性上爆红提示。**

**总结：显然：应该保证的是xml文件不爆红，程序才有可能按照设计的逻辑正确运行，而要让xml文件中的对应的bean不爆红，1是添加property子元素，且在这个子元素中，通过name属性对应上注解的方法； 2是在bean中设置autowire属性，其值只能是byName或者byType。同时在这个xml文件中能正确找到自动配置对应的资源。所以实质上，@Requried只是作为提示工具，防止程序不同于设计逻辑.**

**基于java的配置方式：**

**所谓基于java的配置，资源的显示配置不用再xml文件中写明，立刻java文件中进行配置，以让xml文件中更加简洁，亦或者将不同类型的资源分开书写配置，你需要使用两个注解：@Configuration 和 @Bean；**

**同时你还需要再xml配置文件中如下声明：**

<context:component-scan *base-package*="@Configuration的类注解所在的包"/>

**java类的语法规则：再类定义之前使用@Configuration注解，在方法名之前使用@bean注解；**

**@Configuration ⬄ xml文件的beans元素**

**@Bean ⬄ xml文件中beans元素的子元素bean，而被注解的方法名就是bean的id属性值，方法的返回类型的完全限定名称就是calss属性的值**

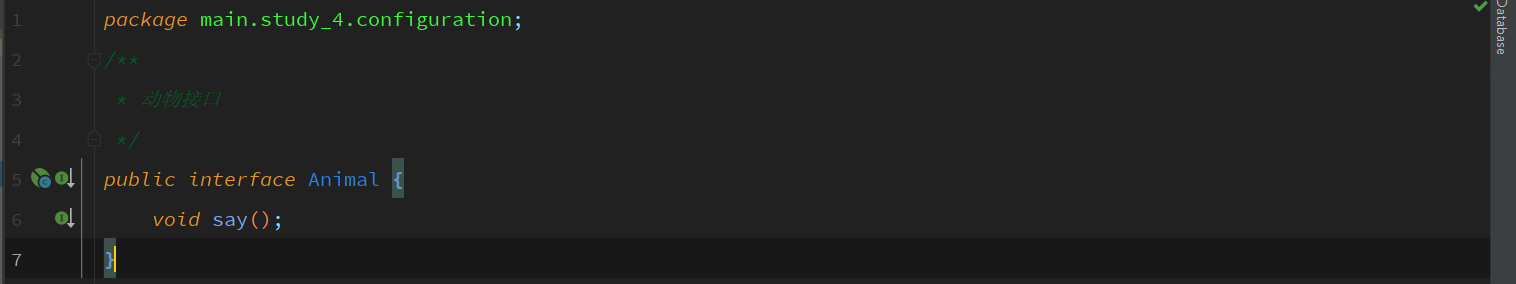
**@Bean注解可以用于任何返回类型是对象的方法前。这个方法的返回值就是xml相当于xml文件中bean所配置的资源。**

**xml问价中的context:component元素会搜索指定包中的所有@Configuration注解，一旦某个类被检索到，该类的所有被@Bean注解的方法都会被依次自动执行，其返回值所代表的对象资源类似于bean元素交付给Spring容器控制。**

**——————————————————————————————————————————————————**

Animal.class:接口

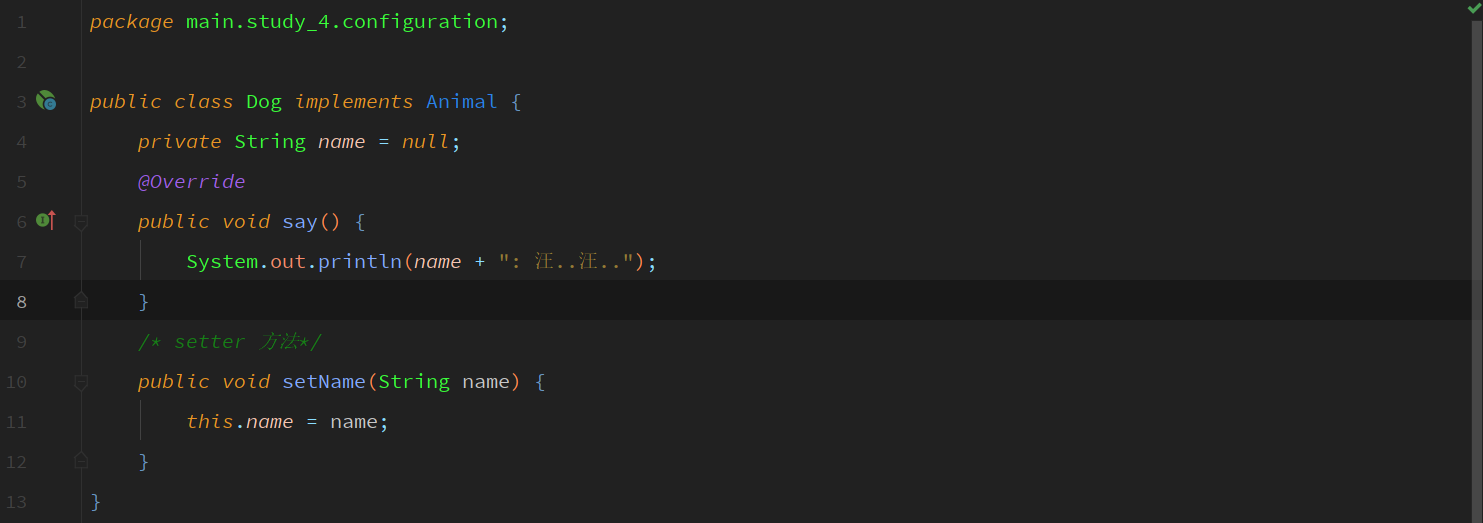
**——————————————————————————————————————————————————**



**——————————————————————————————————————————————————**

Dog.class

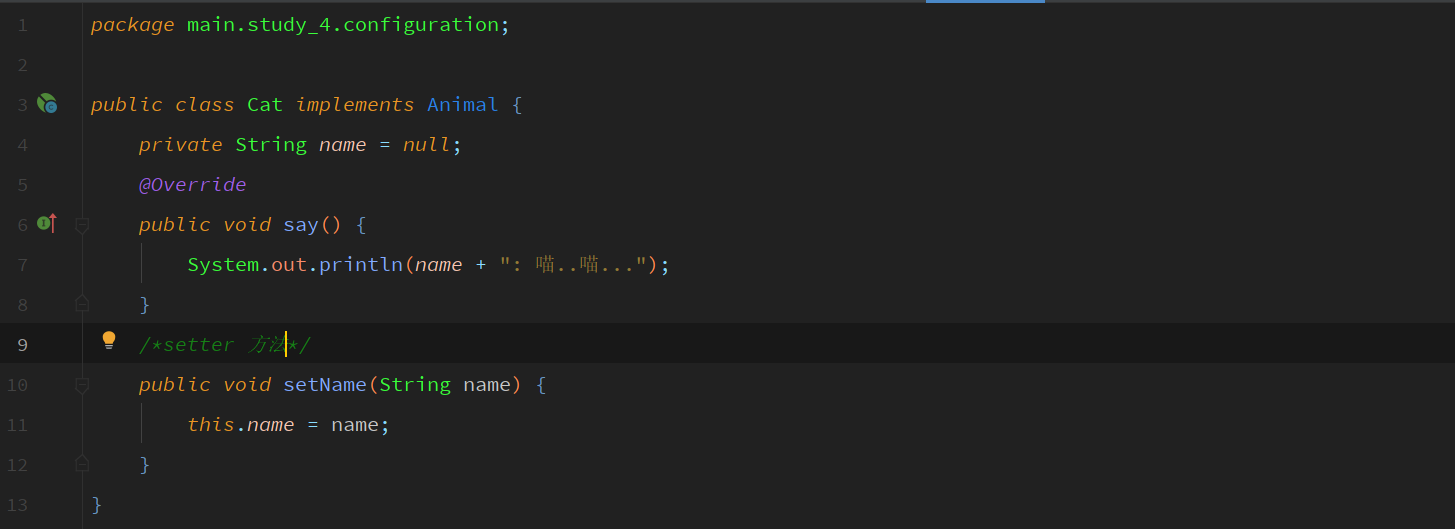
**——————————————————————————————————————————————————**



**——————————————————————————————————————————————————**

Cat.class

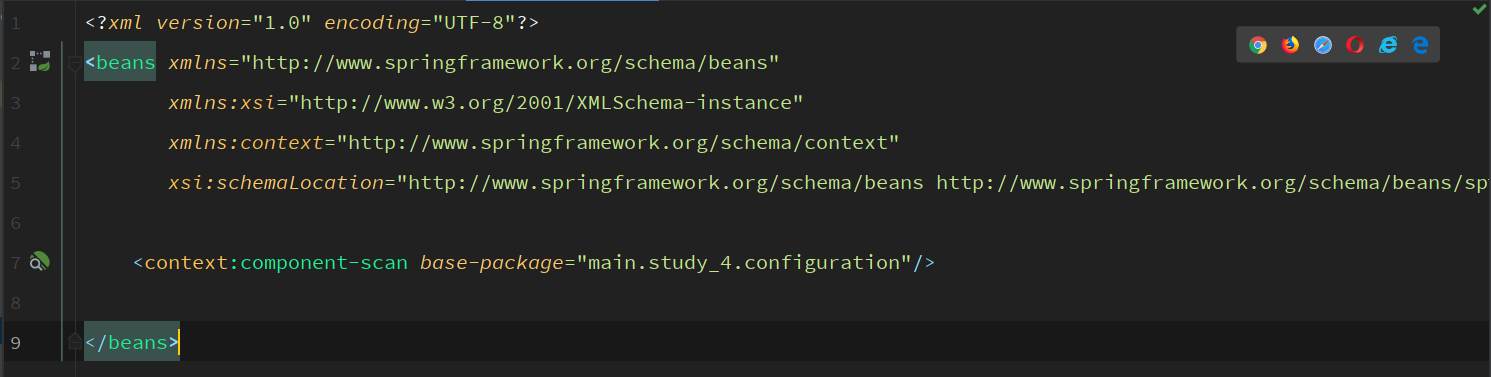
**——————————————————————————————————————————————————**



**——————————————————————————————————————————————————**

Configuration.xml

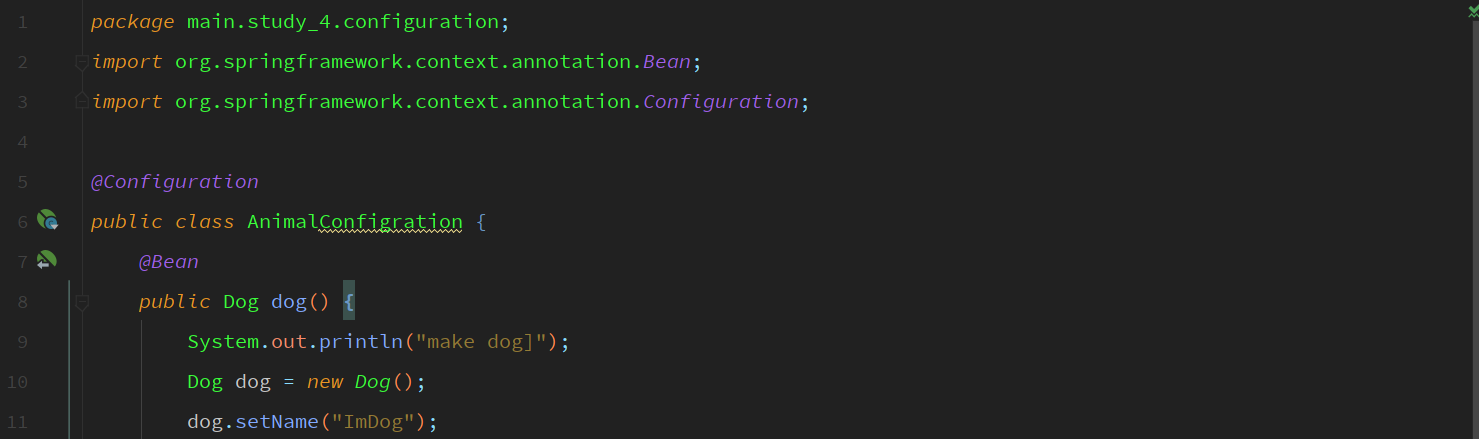
**——————————————————————————————————————————————————**

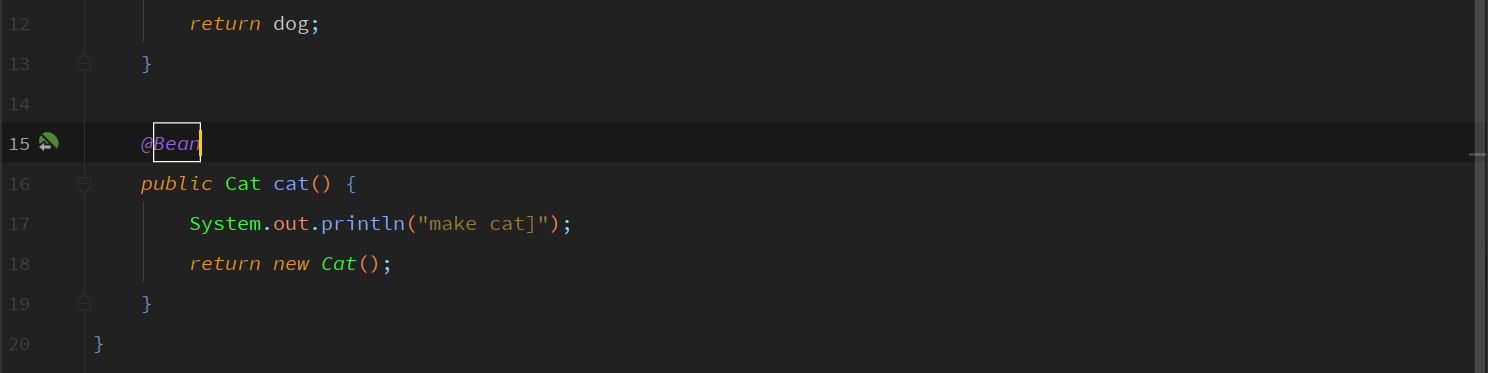


**——————————————————————————————————————————————————**

Configuration.class : 基于java的配置

**——————————————————————————————————————————————————**

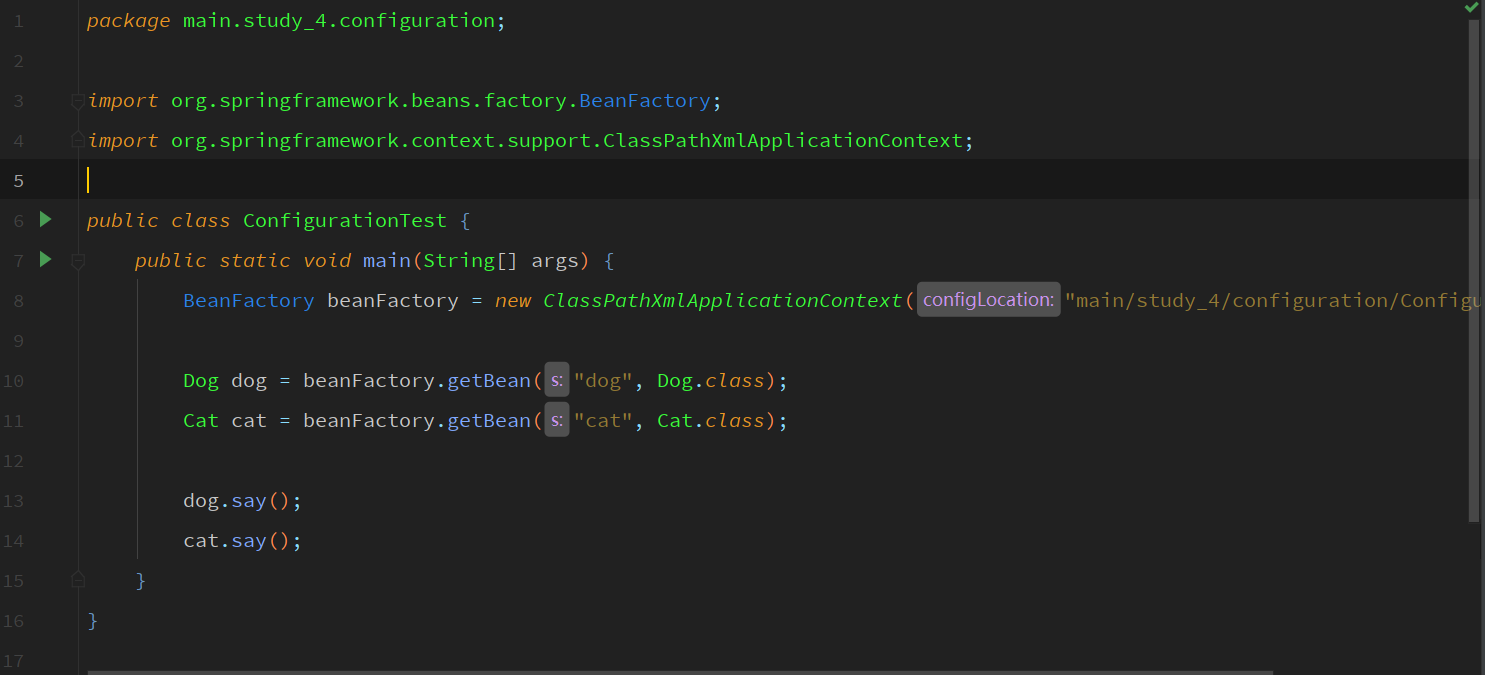




**——————————————————————————————————————————————————**

**ConfigurationTest.class**

**——————————————————————————————————————————————————**



**——————————————————————————————————————————————————**

输出结果：

**——————————————————————————————————————————————————**

