[Spring @Configuration用annotation装配spring](http://blog.csdn.net/tanksyg/article/details/8556769)

分类： [spring](http://blog.csdn.net/tanksyg/article/category/1338051)2013-01-30 16:31 7113人阅读 [评论](http://blog.csdn.net/tanksyg/article/details/8556769#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/tanksyg/article/details/8556769#report)

下面是一个典型的Spring配置文件（application-config.xml）：

**Xml代码**

**[xml]** [view plaincopy](http://blog.csdn.net/tanksyg/article/details/8556769)

1. **<beans>**
2. **<bean** id="orderService" class="com.acme.OrderService"**/>**
3. **<constructor-arg** ref="orderRepository"**/>**
4. **</bean>**
5. **<bean** id="orderRepository" class="com.acme.OrderRepository"**/>**
6. **<constructor-arg** ref="dataSource"**/>**
7. **</bean>**
8. **</beans>**

然后你就可以像这样来使用是bean了：

**java代码**

**[java]** [view plaincopy](http://blog.csdn.net/tanksyg/article/details/8556769)

1. ApplicationContext ctx = **new** ClassPathXmlApplicationContext("application-config.xml");
2. OrderService orderService = (OrderService) ctx.getBean("orderService");

现在Spring Java Configuration这个项目提供了一种通过java代码来装配bean的方案：

**[java]** [view plaincopy](http://blog.csdn.net/tanksyg/article/details/8556769)

1. @Configuration
2. **public** **class** ApplicationConfig {
4. **public** @Bean OrderService orderService() {
5. **return** **new** OrderService(orderRepository());
6. }
8. **public** @Bean OrderRepository orderRepository() {
9. **return** **new** OrderRepository(dataSource());
10. }
12. **public** @Bean DataSource dataSource() {
13. // instantiate and return an new DataSource …
14. }
15. }

然后你就可以像这样来使用是bean了：

**[java]** [view plaincopy](http://blog.csdn.net/tanksyg/article/details/8556769)

1. JavaConfigApplicationContext ctx = **new** JavaConfigApplicationContext(ApplicationConfig.**class**);
2. OrderService orderService = ctx.getBean(OrderService.**class**);

 这么做有什么好处呢?

     1.使用纯java代码，不在需要xml

     2.在配置中也可享受OO带来的好处

     3.类型安全对重构也能提供良好的支持

     4.依旧能享受到所有springIoC容器提供的功能