Spring注解驱动开发第49讲——Spring IOC容器创建源码解析(九)之Spring IOC容器创建源码总结

经过前面8讲的学习，我们就阅读了一下Spring IOC容器整个的创建以及 初始化 过程的源码，在跟踪Spring源码时，整个流程可以说是非常复杂的。因此，在本讲中，我们就来对其做一个小的总结。

首先，我们得需要掌握Spring中的一些核心思想，我们所要掌握的第一个核心思想就是，**Spring IOC容器在启动的时候，会先保存所有注册进来的bean的定义信息，将来，BeanFactory就会按照这些bean的定义信息来为我们创建对象**。

那么，如何来编写这些bean的定义信息呢？你可以有如下两种方式来编写这些bean的定义信息。

1. 使用XML配置文件的方式来注册bean。其实，这种方式说到底无非就是使用 `<bean>` 标签来向IOC容器中注册一个bean的定义信息，这种方式我们已经很熟悉了
2. 使用@Service、@Component、@Bean等等注解来注册bean。其实，这种方式就是使用注解向IOC容器中注册一个bean的定义信息

我们所要掌握的第二个核心思想就是，当IOC容器中有保存一些bean的定义信息的时候，它便会在合适的时机来创建这些bean，而且主要有两个合适的时机，分别如下：

1. 就是在用到某个bean的时候。在统一创建所有剩下的单实例bean之前，有一些bean，比如像后置处理器啦等等这些组件，需要用到它的时候，都会利用getBean方法创建出来，创建好以后便会保存在容器中，以后我们就可以直接从容器中获取了
2. 统一创建所有剩下的单实例bean的时候。相信你应该对此很熟悉了，这不就是我们在跟踪Spring IOC容器创建过程的源码时所分析的一个步骤嘛，即 `finishBeanFactoryInitialization(beanFactory)`，这一步便是来初始化所有剩下的单实例bean的。

   也就是说，所有IOC容器中注册的单实例bean，如果还没创建对象，那么就在这个时机创建出来。

当然了，在整个单实例bean创建的过程中，有最核心的一个思想需要大家掌握，那就是BeanPostProcessor（即后置处理器）。

我们知道，每一个单实例bean在创建完成以后，都会使用各种各样的后置处理器进行处理，以此来增强这个bean的功能。举一个例子，使用@Autowired注解即可完成自动注入，这是因为Spring中有一个专门来处理@Autowired注解的后置处理器，即AutowiredAnnotationBeanPostProcessor。

还记得我们以前在讲述Spring AOP底层原理时，有一个叫AnnotationAwareAspectJAutoProxyCreator的后置处理器吗？如果我要是没记错的话，它的作用就是来为bean来创建代理对象的，通过代理对象来增强这个bean的AOP功能。

这里我只举了以上两个后置处理器为例子，但是，在Spring中其实是有非常多的后置处理器的，它们一般都是在我们bean初始化前后进行逻辑增强的。你现在可以看到Spring中的后置处理器是多么的重要了吧😊，说什么你都得掌握它。

最后，我们所要掌握的第四个核心思想就是，Spring的事件驱动模型。它涉及到了两个元素，分别如下：

1. ApplicationListener：它是用来做事件监听
2. ApplicationEventMulticaster：事件派发器。它就是来帮我们进行事件派发的

以上就是我们 Spring源码 中的一些比较核心的思想。对于我们而言，最重要的是需要理解与掌握后置处理器，因为Spring都是利用各种各样的后置处理器来对bean进行增强处理的。除此之外，你还得理解Spring中的事件驱动模型。

至此，对于Spring IOC容器创建源码的分析，我们就总结完了。希望能对读者有所帮助。完结撒花~~~