1. 1111Spring源码整体认知

Ioc里面管理者很多的bean，如何将类转换成bean进行管理？

1. 配置类 xml @注解 javaconfig
2. 加载spring上下文
3. Xml:new ClassPathApplicationConfig(“xnl”)
4. @:new AnnotationConfihApplicationContext(config.class)
5. getBean()

右边的图

**2、BeanFactory是spring的顶层接口，使用了简单的工厂模式，负责生产Bean,只是封装生产bean的那个模块**

**左边的图**

**3、BeanDefinitionReader（销售）先读取配置类，BeanDefinitionScanner再去扫描有加@component要放入容器的类，然后再BeanDefinitionRegistry再注册到BeanDefinitionMap容器里面，然后将统一的图纸BeanDefinition交给BeanFactory,只有ApplicationContext有这些功能，但是BeanFactory只有单纯的生产bean的功能**

**4、右边的图 bean 的加载过程**

**bean最后会放到一个map里面，单例池，一级缓存，然后getBean到这个zmap里面去取**

**bean的生命周期：填充属性的时候，存在循环依赖问题**

**bean的生命周期：在实例化的时候，反射是spring工厂控制的，而工厂方法@bean是自己new的，可以在里面自由的控制bean实例**

**扩展点：左图：beandefinition的时候，beafactoryPostProcessor（bean的工厂的后置处理器）可以获得bean定义然后修改bean定义，beandefinitionregistryPostProcessor注册bean定义**

**除了ioc，别的都是靠扩展点实现**

**扩展点：右图：beanpostProcessor（bean 的后置处理器），bean’的生命周期过程可以调用9次的beanpostProcessor**

**Aop：在bean生命周期结束后**

**Bean的生命周期：实例化（反射） 填充属性 初始化（里面会调用很多XXXAware） 其中有不少的beanPostProcessor**

**总结：类加载到ioc，先加载beanDefinition（有这些步骤：先扫描配置类beanDefinitionReader,然后读取合格的类（比如加了componnet，resources这些注解的类，）BeanDefinitionScanner,）然后注册beanDefinition的BeanDefinitionRegistry，其中的扩展点 BeanFactoryPostProcessor修改beanDefinition，beanDefinitionRegistryPostProcessor用来注册beanDefinition**

**beanFactory#getBean,既可以生产又可以读取**

**mybatis实现了beanDefinitionRegistryPostProcessor接口（注册BeanDefinition，）**

**beanfactory就是生产bean’的，而applicationContex再次基础上，还多了扩展点**