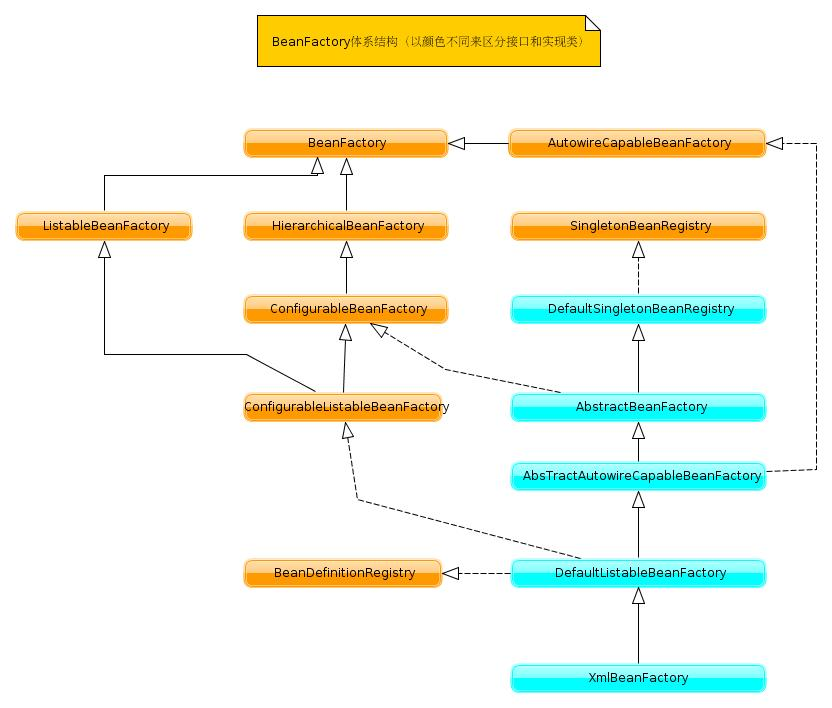
1. Spring通过反射的方式在IOC容器 中创建bean，所以bean中要有无参的构造器。
2. Bean配置形式：基于XML方式和注解的方式。 Bean的配置方式：全类名（反射方式），工厂方法（静态工厂方法&实例工厂方法），FactoryBean
3. 使用构造器注入属性值可以指定参数的位置和参数的类型来区分重载的构造器。
4. 属性值也可以用value子节点进行赋值，如果包含特殊字符则用![CDATA[]]包含起来。
5. 可以使用ref属性建立bean之间的关系，可以在要引用bean的地方创建一个bean，称内部bean，内部bean不能被外面引用，此事可以省略ref。
6. 属性要为null时要写成<null/>
7. Spring也可以为级联属性赋值，但是属性需要先初始化后才能赋值。
8. Spring配置集合属性时可以用list，set，map节点。Map配置需要加上entry子节点。
9. 可通过p命名空间为bean的属性赋值。
10. Spring可以根据autowire属性进行自动装配bean。有byName和byType两种方式装配bean，byName根据bean的名字和当前bean的setter风格的属性进行自动装配，若有匹配的，则进行自动装配，若没有，则不装配。byType根据bean的类型和当前bean的属性的类型进行自动装配，若IOC容器有1个以上的类型匹配的bean，则抛出异常。Autowire装配的是引用类型的属性。
11. Spring可以通过parent属性继承一个bean,子bean可以覆盖父bean的属性。
12. Bean的abstract属性为true的bean不用被IOC实例化，只能被继承。
13. 如果bean的class属性没有指定的话，则必须把abstract属性设为true。
14. Spring可以通过depends-on属性配置bean之间的依赖关系，被依赖的bean要在本bean实例化之前实例化。
15. Spring可以通过scope属性配置bean的作用域，默认值是singleton（既每次调用都是同一个bean）。其他取值：prototype，request，session。Prototype是指原型的，容器初始化时不创建bean的实例，而在每次请求时创建一个新的实例。
16. Spring可以使用<context:property-placeholder/>导入外部属性文件，<context:property-placeholderlocation= “classpath:\*.properties”/>,例如读取数据库等配置信息。
17. 在spEL中引用类的今天属性或者方法用T()。
18. IOC容器对bean的生命周期管理过程：
19. 通过构造器或工厂方法创建bean的实例
20. 为bean的实例设置值和其他bean的引用
21. 调用bean的初始化方法
22. 使用bean
23. 当容器关闭时，调用bean的销毁方法 。

在bean的声明中可以设置init-method和destroy-method属性为bean指定初始化方法

1. Bean的后置处理器允许调用初始化方法前后对bean进行额外的处理，需要实现BeanPostProcessor接口，在配置中不需要指定id属性。后置处理器是对IOC容器里的所有bean实例逐一处理，而非单一实例。
2. Spring还可以通过FactoryBean来配置Bean，自定义的FactoryBean需要实现Spring提供的FactoryBean接口，配置的FactoryBean实际返回的是FactoryBean的getObject（）方法返回的实例。
3. <context:component-scan/>的使用：
4. 在组件类上使用了特定的注解之后，就要在配置文件中声明。
5. Base-package属性指定了一个需要扫描的基类包，IOC容器就会扫描这个基类包及其子包中的所有类，如需扫描多个包，则用逗号分开。
6. User-deafult-filter表示使用默认的filter（所有注解），默认true，配合下面两个子节点使用。
7. 如果希望扫描特定的类，可用resource-pattern过滤，如：resource-pattern=“autowire/\*.class”
8. <context:include-filter>子节点表示要包含的目标类
9. <context:exclude-filter>子节点表示排除在外的目标类
10. 以上子节点选择类的类型有两种：注解方式和类名或者接口名，即type=“annotation”或type=“assinable”。
11. 默认情况下使用@Authwired注解的属性都需要被设置，注解的属性都需要被设置，则设置注解的required属性为false。
12. 当IOC容器里存在多个类型兼容的bean时，通过类型的自动装配将无法工作，此时可以在@Qualifier注解中提供名称参数。
13. Spring提供类相应的Resource资源类,可以通过资源地址前缀类加载不同类型的资源

如：classpath:和classpath\*: (从类路经下加载，前者只会扫描第一个包，后者会扫描所有包)

file: (从文件系统中加载)，http:// (从web服务器中加载) ftp： (从ftp服务器中加载)

25.BeanFactory的类体系结构，BeanFactory主要方法是getBean()方法

ListableBeanFactory:定义了访问容器中Bean基本信息的若干方法，如查看Bean个数，获取Bean的配置名等

HierarchicalBeanFactory:父子级联Ioc容器的接口

ConfigurableBeanFactory:是一个重要的接口，增强了IOC容器的可定制性，定义了设置类装载器，属性编辑器，容器初始化等方法

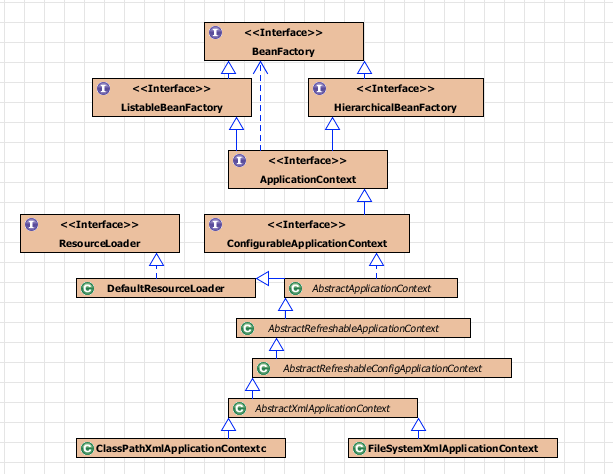
AutowireCapableBeanFactory:定义了容器中Bean按某种规则进行自动装配的方法

SingletonBeanRegistry：定义了允许在运行期间向容器中注册单个实例Bean的方法

26. 对于单实例的Bean来说，BeanFactory会缓存Bean实例，缓存器是用HashMap实现的。当第二次获取Bean时直接从Ioc容器的缓存中获取。

27.ApplicationContext的主要实现类是ClassPathXmlApplicationContext和FileSystemXmlApplicationContext

28.ApplicationContext体系结构

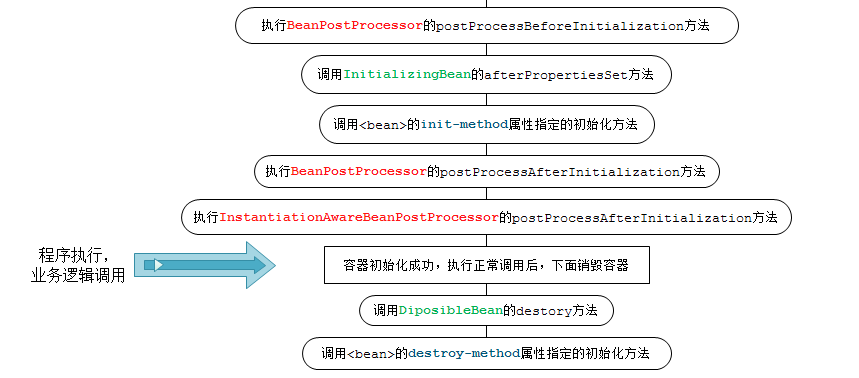
ApplicationEventPublisher:让容器拥有发布应用上下文事件的功能，包括容器启动事件，关闭事件。

MessageSource:为应用提供i18n国际化消息访问的功能

LifeCycle:提供了start()和stop()两个方法，主要用于控制异步处理过程。

28. ApplicationContext的初始化和BeanFactory有个重大的区别:BeanFactory在初始化容器时，并未实例化Bean,直到第一次访问某个Bean时才实例华Bean;而ApplicationContext则在初始化应用上下文时就实例化所有bean。两者最大不同之处：ApplicationContext会利用反射机制自动识别配置文件中定义的BeanPostProcessor,InstantiationAwareBeanPostProcessor和BeanFactoryPostProcessor，并自动将它们注册到应用上下文中;而后者需要手工调用addBeanPostProcessor()方法进行注册。

29.Bean生命周期



Bean的完整生命周期经历了各种方法调用，这些方法可以划分为以下几类：

Bean自身的方法：这个包括了Bean本身调用的方法和通过配置文件中<bean>的init-method和destroy-method指定的方法

Bean级生命周期接口方法：这个包括了BeanNameAware、BeanFactoryAware、InitializingBean和DiposableBean这些接口的方法

容器级生命周期接口方法：这个包括了InstantiationAwareBeanPostProcessor 和 BeanPostProcessor 这两个接口实现，一般称它们的实现类为“后处理器”。

工厂后处理器接口方法：这个包括了AspectJWeavingEnabler, ConfigurationClassPostProcessor, CustomAutowireConfigurer等等非常有用的工厂后处理器接口的方法。工厂后处理器也是容器级的。在应用上下文装配配置文件之后立即调用。

30. Spring依赖注入的方式有：属性注入(setter方法)，构造函数注入，工厂方法注入(非静态和静态)

31. 内部Bean会忽略id,name,scope属性

32.注入null值时，value为”<null/>”