Spring IoC容器

1. BeanFactory和ApplicationContext的关系

Spring提供的一个核心功能就是IoC容器，帮助用户管理各个Bean的生命周期。而BeanFactory是所有Spring的IoC容器的父接口，其中定义了作为一个容器所必须有的基本功能；ApplicationContext则是通过继承抽象BeanFactory类，实现接口等方式，在拥有BeanFactory的IoC容器的基本功能的同时，还有许多其他的高级功能，可以说是一个基于BeanFactory的更加高级的IoC容器。

一般来说，ApplicationContext会通过持有一个DefaultListableBeanFactory对象，来拥有IoC容器的基本功能。

\* 从类的名字上，就可以看出一个ApplicationContext在基本功能的基础上，又扩展了哪些额外的功能，比如FileSystemXmlApplicationContext，就是支持从文件系统中读取xml配置文件

2. BeanFactory接口含有的功能

(1) 各种获取Bean的方法：

Object getBean(String name); 通过名字获取Bean

<T> T getBean(String name, Class<T> requiredType); 只获取指定类型的Bean

Object getBean(String name, Object… args); 获取Bean时提供自定义的构造参数

(2) 检查是否含有特定的Bean:

Boolean containsBean(String name);

(3) 检查是否是单例/多例：

Boolean isSingleton / isPrototype(String name);

(4) 获取指定Bean的类型信息：

Class getType(String name);

3. Resource类

Resource类已经它的子类，是Spring中用于封装I/O的类，不同的Resource子类，可以获取不同地方的资源文件，比如FileSystemResource从文件系统中定位资源，ClassPathResource从类路径中定位资源等。

4. 建立IoC容器的基本过程

(1) 以Resource类的形式指定IoC容器配置文件的抽象资源，其中包含了BeanDefiniton的信息

(2) 根据不同的需求，创建一种BeanFactory

(3) 创建一个资源文件的读取器，通过读取指定的资源文件，生成一个个BeanDefinition并存入BeanFactory中，比如使用XmlBeanDefinitonReader读取执行的xml格式的配置文件

(4) 将解析得到的BeanDefinition存放入容器的一个HashMap中，这个时候用户就可以直接使用这个IoC容器了

5. Bean和BeanDefinition的区别

Bean定义的载入和依赖注入是两个独立的过程，在建立IoC容器的过程中，存入的BeanDefinition并不是Bean本身，而是Bean的定义，只有在用户第一次调用getBean方法时，才会通过根据这个Bean的定义，通过反射的方法生成这个对象。当然也可以在BeanDefinition中设置lazy\_init的属性为真，那么在读取Bean定义的时候，就会生成相应的对象了。