1. BeanFactory：提供getBean方法的接口类
2. AbstractBeanFactory：

继承BeanFactory并实现getBean()方法。

继承DefaultSingletonBeanRegistry，由Registry的getSingleton()方法实现getBean()，即从ObjectMap中取出对象。

* 1. AbstractBeanFactory实现的方法：getBean()、doCreateBean()、addBeanPostProcessor()、getBeanPostProcessors()。
  2. AbstractBeanFactory抽象的方法：createBean()、getBeanDefinition(beanName)

1. AbstractAutowireCapableBeanFactory：主要方法是createbean()，继承AbstractBeanFactory。它的createBean方法依赖于从BeanDefinition获取beanClass的对象。
   1. createBean与AutowireBeanFacotry：createBean从BeanClass中获取构造器列表，并根据构造入参选择对应的构造器，之后使用实例化接口(InstatiationStrategy)来实例化。
   2. InstatiationStrategy：主要使用到beanClass和Constrcuctor，有cglib和jdk两种实例化策略。
   3. applyPropertyValue：用以属性填充，从BeanDefinition中获取到PropertyValues。
2. PropertyValue：典型Key-Value对象，可用于接收属性填充的对象，name为属性名，value为值。
3. PropertyValues：内置PropertyValue列表，getPropertyValue可根据名字获取PropertyValue。
4. BeanReference：若PropertyValue的value属性为BeanReference对象，则表明该属性为需要注入的引用对象，注入时会使用getBean()的方法来获取Bean。
5. DefaultListableBeanFactory：继承AbstractBeanFactory并实现BeanDefinitionRegistry。属性BeanDefinitionMap用以注册和获取BeanDefinition。
6. Resouce：其提供的getInpuStream()方法用于获取资源的输入流InputStream。输入流有Url、classPath、FileSystem这3种来源，入参分别为Url、Path(String)、File。
7. ResourceLoader：Resource的封装使用接口，只需要传递location(String)便可以获取Resource。DefaultResourceLoader会把3种方式都尝试一遍以加载资源。
8. BeanDefinitionReader：提供获取BeanRegistry和ResourceLoader方法的get接口，以及加载BeanDefinition的方法loadBeanDefinition()。AbstractBeanDefintion实现getBeanRegistry和getResourceLoader接口。
9. XmlBeanDefinitionReader：doLoadBeanDefintion从xml的输入流(InputStream)中加载BeanDefinition。
10. BeanFactoryPostProcessor：允许在注册BeanDefinition后但未实例化之前(createBean)修改BeanDefinition的信息。
11. BeanPostProcessor：在bean实例化(createBean)之后修改或替换Bean对象，即在初始化方法执行前后修改Bean。即BeforeIntialization、AfterInitialization。

13.1 实例化即使用构造器创建对象，对象被创建出来后还可以执行自定义的初始化方法。

1. 继承关系：

- ListableBeanFactory(getBeansOftype)

-- ApplicationContext

--- ConfigurableApplicationContext(refresh、registerShutdownHook、clsoe)

---- AbstractApplicationContext(实现refresh)

1. 继承关系：

* ListableBeanFactory(getBeansOfType)
* ConfigurableBeanFactory(addBeanPostProcessor)
* AutowireCapableBeanFactory(applyBeanPostProcessor+Before/After+Initialization)执行BeanPostProcessors方法。

-- ConfigurableListableBeanfactory

(PreInstantiate、addBeanPostProcessor)

14. 销毁方法原理：向JVM注册ConfigurableBeanFactory的销毁函数。

14.1 DisposableBeanAdapter: 如果组合的bean不为DisposableBean，则执行BeanDefinition定义的销毁方法。如果组合的bean为DisposableBean，则先执行它的destory方法，最后如果BeanDefinition另外定义了不为”destroy”的销毁，则执行该方法。