**CAP12 BeanFactory的两个重要后置处理器**

**1，扩展原理－BeanFactoryPostProcessor**

BeanFactoryPostProcessor：beanFactory的后置处理器；

作用如下：

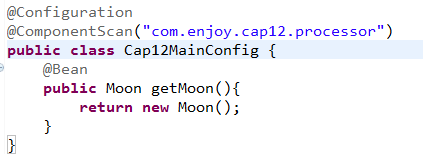
a),在BeanFactory标准初始化之后调用，来定制和修改BeanFactory的内容；

b),所有的bean定义已经保存加载到beanFactory，但是bean的实例还未创建

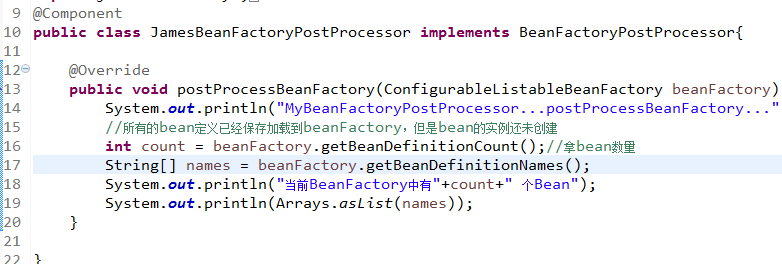
**注意：**之前也讲过BeanPostProcessor，它是bean后置处理器，bean创建对象初始化前后进行拦截工作的

操作步骤：

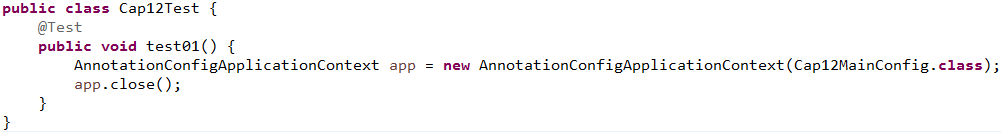
1〉新建ExtConfig.java配置类



新建JamesBeanFactoryPostProcessor.java处理器类（在com.enjoy.cap12.processor目录下）

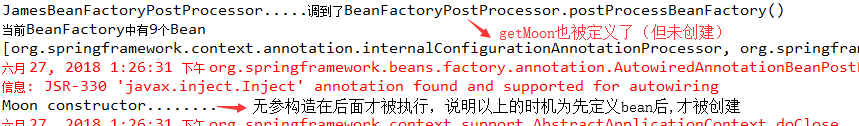


BeanFacotry是在bean组件创建之前还是之后生成的呢？写一个测试用例



不难发现，在“Moon constructor”创建之前，当前9个bean已被加载到beanFactory中了。

测试结果结下：



那么以上BeanFactoryPostProcessor执行原理是怎么样的呢，打断点F5调试一下:



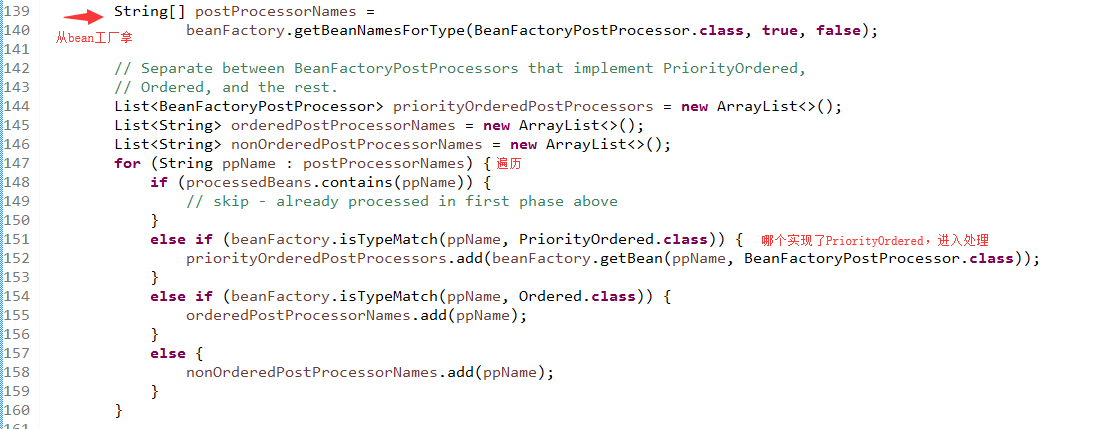
跟踪debug栈,不难发现以下步骤如下：

1)、ioc容器创建对象

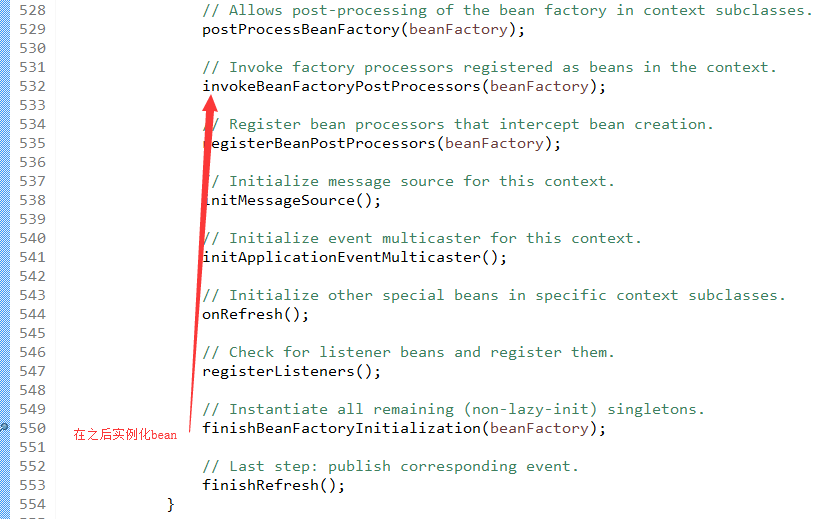
2)、invokeBeanFactoryPostProcessors(beanFactory);

如何找到所有的BeanFactoryPostProcessor并执行他们的方法；

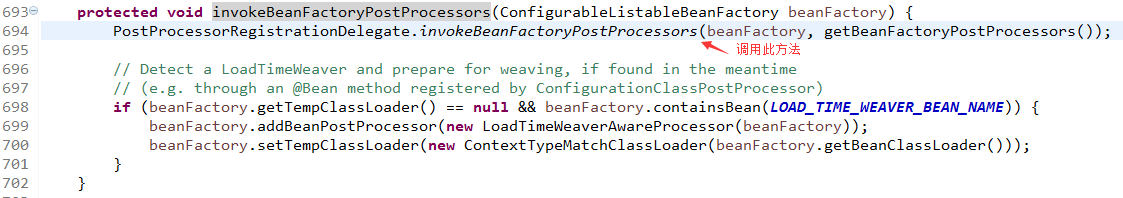
a）、直接在BeanFactory中找到所有类型是BeanFactoryPostProcessor的组件，并执行他们的方法



\* b）、在初始化创建其他组件前面执行



然后再点进去,



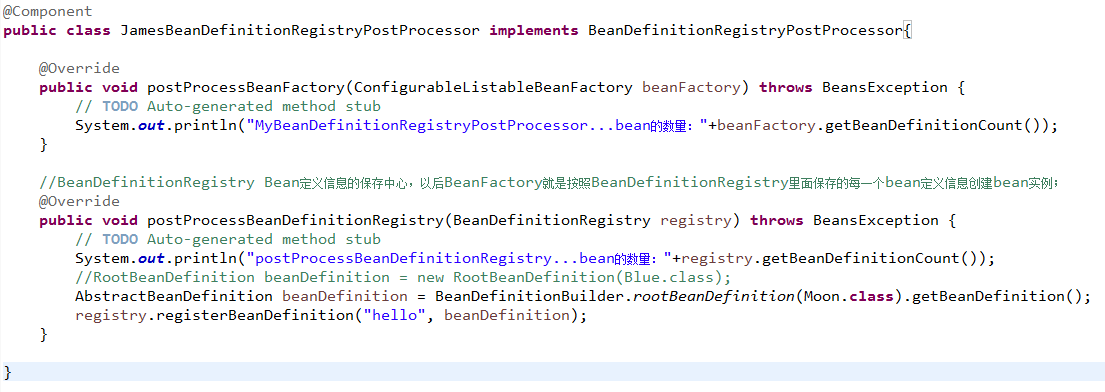
**2，扩展原理－BeanDefinitionRegistryPostProcessor**

postProcessBeanDefinitionRegistry();在所有bean定义信息将要被加载，bean实例还未创建的；

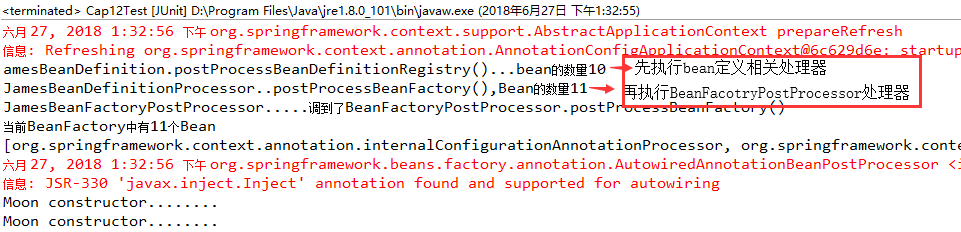
操作步骤：

新建JamesBeanDefinitionRegistryPostProcessor

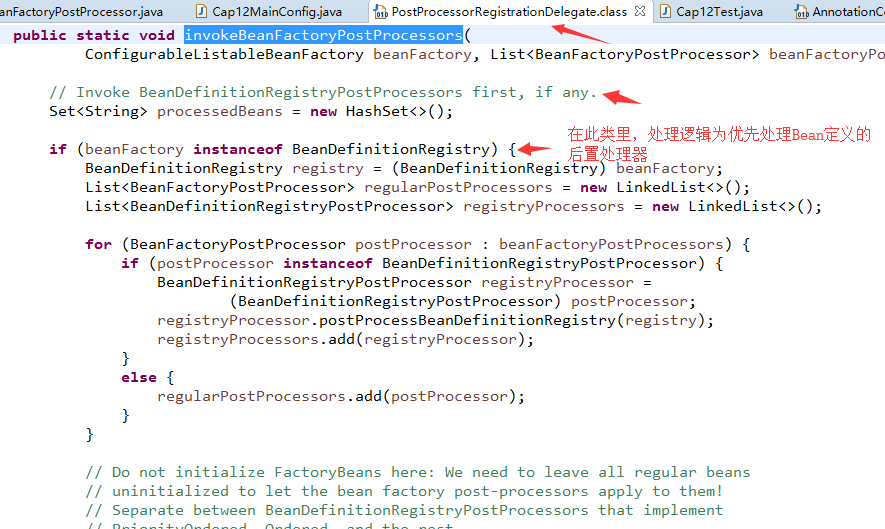
**注意**：可使用BeanDefinitionBuilder构建器来创建bean定义信息



测试结果如下：

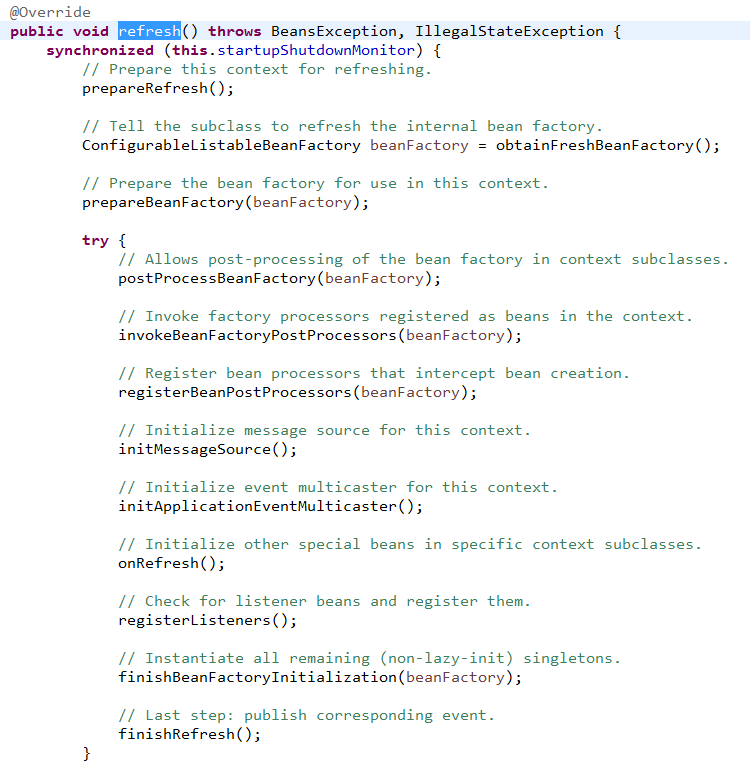


源码分析：从refresh()-->invokeBeanFactoryPostProcessors()-->PostProcessorRegistrationDelegate.invokeBeanFactoryPostProcessors(), 跟进去即发现代码逻辑如下：



所以，在任何情况下都会优先执行BeanDefinitionRegistryPostProcessor的处理器，而BeanFactoryPostProcessor的处理器在它后面执行

**3，IOC容器处理流程（其实就是研究一下refresh()下的这些方法）**



Spring容器的refresh()【创建刷新】;

1、prepareRefresh()刷新前的预处理;

1）、initPropertySources()初始化一些属性设置;子类自定义个性化的属性设置方法；

2）、getEnvironment().validateRequiredProperties();检验属性的合法等

3）、earlyApplicationEvents= new LinkedHashSet<ApplicationEvent>();保存容器中的一些早期的事件；

2、obtainFreshBeanFactory();获取BeanFactory；

1）、refreshBeanFactory();刷新【创建】BeanFactory；

110行：创建了一个this.beanFactory = new DefaultListableBeanFactory();

设置id；

2）、getBeanFactory();返回刚才GenericApplicationContext创建的BeanFactory对象；

3）、将创建的BeanFactory【DefaultListableBeanFactory】返回；

3、prepareBeanFactory(beanFactory);BeanFactory的预准备工作（以上创建了beanFactory,现在对BeanFactory对象进行一些设置属性）；

1）、设置BeanFactory的类加载器、支持表达式解析器...

2）、添加部分BeanPostProcessor【ApplicationContextAwareProcessor】

3）、设置忽略的自动装配的接口EnvironmentAware、EmbeddedValueResolverAware、xxx；

4）、注册可以解析的自动装配；我们能直接在任何组件中自动注入：

BeanFactory、ResourceLoader、ApplicationEventPublisher、ApplicationContext

5）、添加BeanPostProcessor【ApplicationListenerDetector】

6）、添加编译时的AspectJ；

7）、给BeanFactory中注册一些能用的组件；

environment【ConfigurableEnvironment】、

systemProperties【Map<String, Object>】、

systemEnvironment【Map<String, Object>】