bean中获取spring上下文方式：（基于ApplicationContextAware）

作者：SPPan

原文链接：https://ld246.com/article/1471346495430

来源网站：链滴

<pre class="prettyprint lang-java">

<p>

<pre class="prettyprint lang-java">ApplicationContextAware是在org.springframework.contex包下的一个接口，用于获取spring的上下文，顾名思义，可用通过这个接口获取到spring的上下文context，通俗的讲，就是能通过实现这个接口来获取到spring的IOC容器中的各个bean，具体实现如下步。

```
/** *ApplicationContextAware 接口定义
 */
public abstract interface org.springframework.context.ApplicationContextAware {
  public abstract void setApplicationContext(org.springframework.context.ApplicationContextontext) throws org.springframework.beans.BeansException;
}
```

1、通过这个接口的定义可以发现，实现本接口需要重写setApplicationContext方法。
例如：

```
public class SMSContextInit  implements ApplicationContextAware {
    private static ApplicationContext applicationContext;
    public ApplicationContext getApplicationContext() {
        return applicationContext;
    }
    //重写接口中的方法
    public void setApplicationContext(ApplicationContext aContext) {
        applicationContext = aContext;
    }
    /**
     * 通过beanId得到factory中的bean实例
     * @param beanId
     * @return Object
     */
    public static Object getBean(String beanId) {
        Object obj = null;
        if (applicationContext != null)
            obj = applicationContext.getBean(beanId);
        return obj;
    }
}
```

2、在spring的配置文件中需要配置实现类给spring管理。

```
&lt;bean class="gnnt.MEBS.integrated.mgr.utils.SMSContextInit" /&gt;
```

通过实现的过程不难发现，ApplicationContextAware获取上下文的步骤就是，先把实现类SMSContxtInit 交给spring管理，是spring容器发现SMSContextInit 是ApplicationContextAware的实现类则调用setApplicationContext方法把上下文的引用交给SMSContextInit 类来使用。

另说：
spring虽然提供了ApplicationContextAware接口来使用，但是从耦合性的层面上来看，实现这个接相当于把实现类和spring框架严格的绑定在一起了，有违低耦合的思想。其实要实现低耦合的获取spring上下文也是可以实现的，ApplicationContextAware其实就是使用的setter注入的思想，那么获取下文的操作同样可以用setter注入的方式来实现，代码改写如下：

```
public class SMSContextInit {
    private static ApplicationContext applicationContext;
    public ApplicationContext getApplicationContext() {
        return applicationContext;
    }
    //setter方法用来实现context的注入
    public void setApplicationContext(ApplicationContext aContext) {
        applicationContext = aContext;
    }
    /**
     * 通过beanId得到factory中的bean实例
     * @param beanId
     * @return Object
     */
    public static Object getBean(String beanId) {
        Object obj = null;
        if (applicationContext != null)
            obj = applicationContext.getBean(beanId);
        return obj;
    }
}
```

</pre>

</p>

</pre>