12-07 Spring源码编译和xml解析

1、spring为什么学

1. 其他框架用到了，比如：如果使用netty,会使用到spring
2. 代码更加优雅，体现程序员功力，不喜欢屎山
3. 面试需要

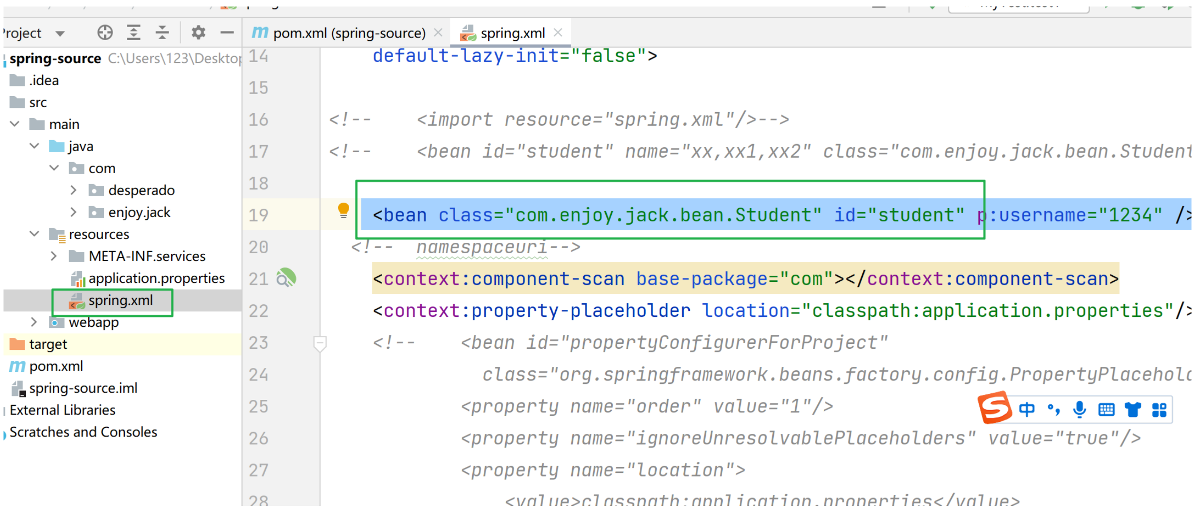
2、编译-代码

spring源码工程中也用到了 命令 bat

<spring.version>5.2.8.RELEASE</spring.version> ------源码版本

3、代码分析

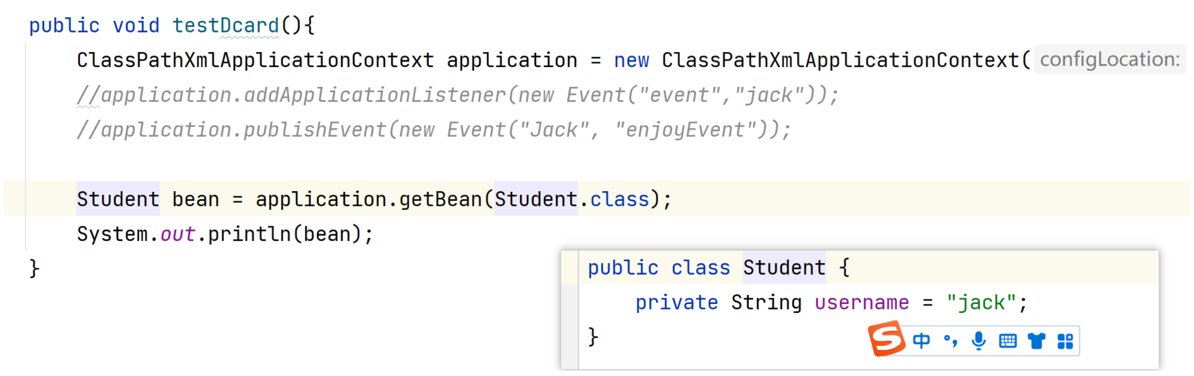
http://117.33.237.52:20070/desperado-image/xml%E8%A7%A3%E6%9E%90%E5%92%8CBeanDefinition%E5%B0%81%E8%A3%85%E6%A0%B8%E5%BF%83%E6%96%B9%E6%B3%95%20refreshBeanFactory().jpg



```
    default-lazy-init="false">

<!--    <import resource="spring.xml"/>-->
<!--    <bean id="student" name="xx,xx1,xx2" class="com.enjoy.jack.bean.Student

    <bean class="com.enjoy.jack.bean.Student" id="student" p:username="1234" />
  <!--  namespaceuri-->
    <context:component-scan base-package="com"></context:component-scan>
    <context:property-placeholder location="classpath:application.properties"/>
    <!--    <bean id="propertyConfigurerForProject"
              class="org.springframework.beans.factory.config.PropertyPlacehol
            <property name="order" value="1"/>
            <property name="ignoreUnresolvablePlaceholders" value="true"/>
            <property name="location">
                <value>classpath:application.properties</value>
```



```
public void testDcard(){
    ClassPathXmlApplicationContext application = new ClassPathXmlApplicationContext( configLocation:
    //application.addApplicationListener(new Event("event","jack"));
    //application.publishEvent(new Event("Jack", "enjoyEvent"));

    Student bean = application.getBean(Student.class);
    System.out.println(bean);
}
```

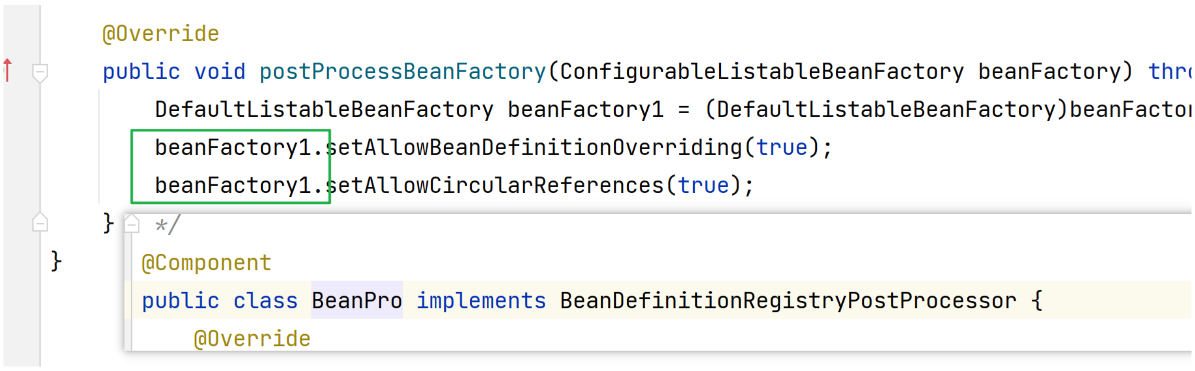
```
public class Student {
    private String username = "jack";
}
```

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${spring.version}</version>
</dependency>
```

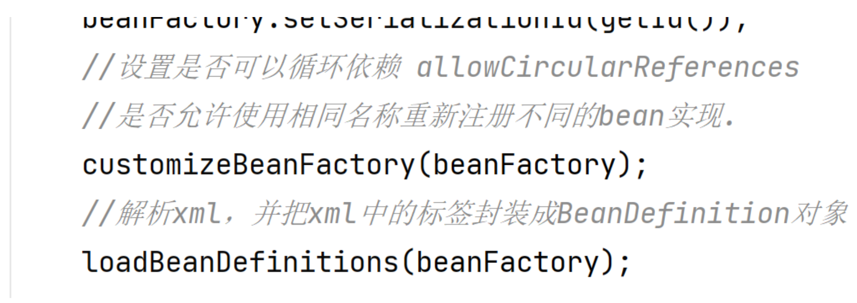
========================

模板设计模式

钩子方法。com.enjoy.jack.designPattern.template



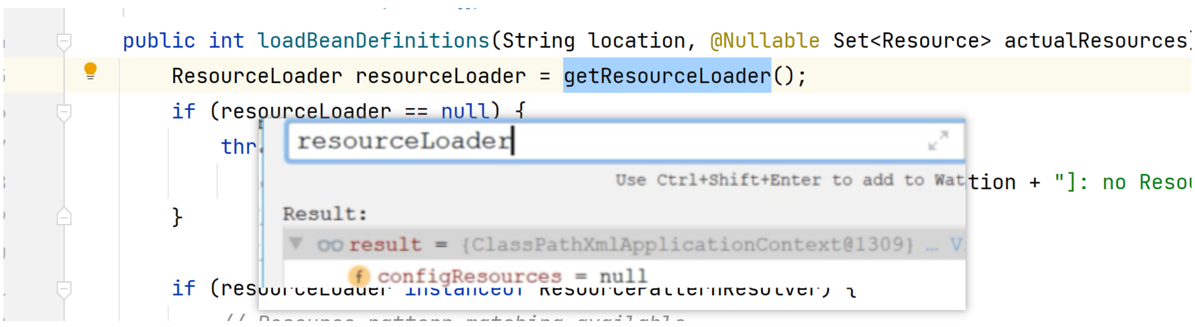
```
    @Override
    public void postProcessBeanFactory(ConfigurableListableBeanFactory beanFactory) thr
        DefaultListableBeanFactory beanFactory1 = (DefaultListableBeanFactory)beanFacto
        beanFactory1.setAllowBeanDefinitionOverriding(true);
        beanFactory1.setAllowCircularReferences(true);
    } */
}
 @Component
 public class BeanPro implements BeanDefinitionRegistryPostProcessor {
     @Override
```



```
beanFactory.setSerializationId(getId());
//设置是否可以循环依赖 allowCircularReferences
//是否允许使用相同名称重新注册不同的bean实现.
customizeBeanFactory(beanFactory);
//解析xml，并把xml中的标签封装成BeanDefinition对象
loadBeanDefinitions(beanFactory);
```

委托设计模式

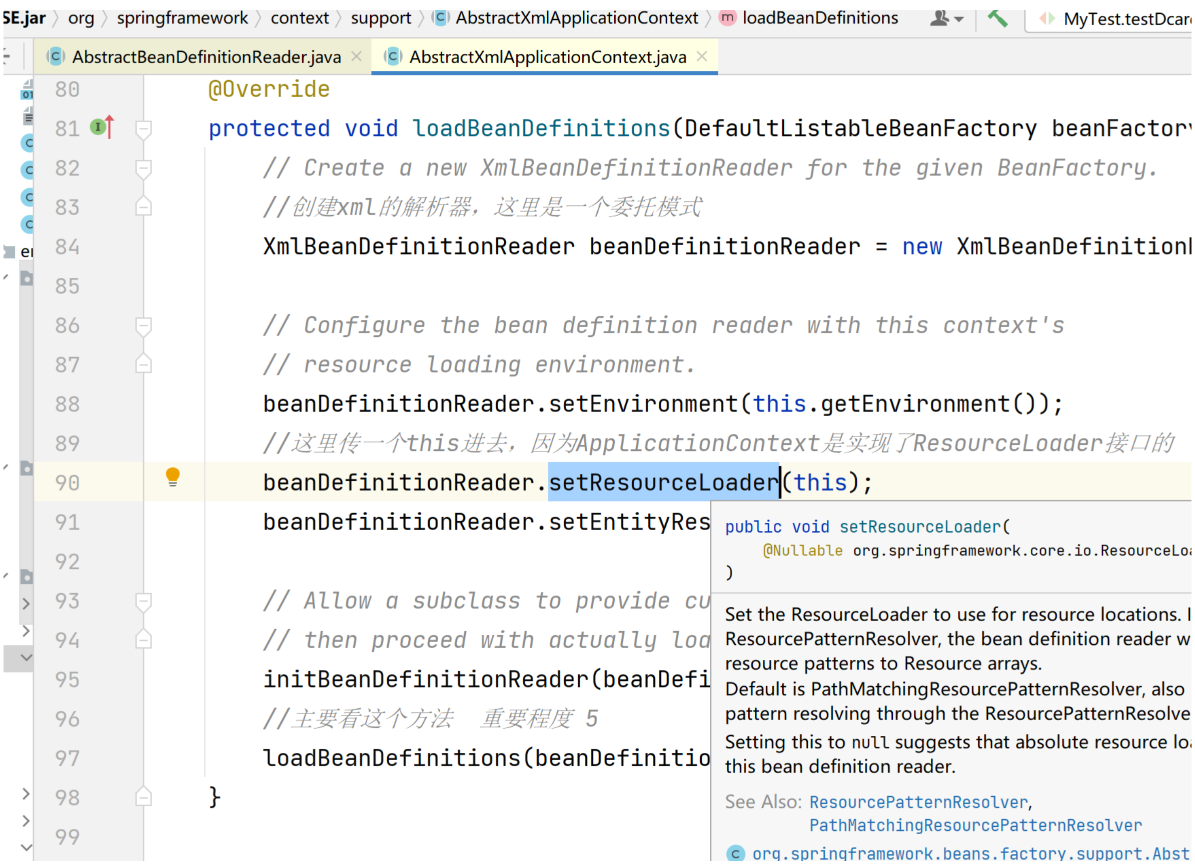
com/enjoy/jack/designPattern/entrust



```
public int loadBeanDefinitions(String location, @Nullable Set<Resource> actualResources
    ResourceLoader resourceLoader = getResourceLoader();
    if (resourceLoader == null) {
        thr
                                                            tion + "]: no Reso
    }

    if (res
```

```
resourceLoader
Use Ctrl+Shift+Enter to add to Wat
Result:
▼ oo result = {ClassPathXmlApplicationContext@1309} … V
    f configResources = null
```



SE.jar 〉 org 〉 springframework 〉 context 〉 support 〉 AbstractXmlApplicationContext 〉 loadBeanDefinitions   MyTest.testDcar

AbstractBeanDefinitionReader.java × AbstractXmlApplicationContext.java ×

```
    @Override
    protected void loadBeanDefinitions(DefaultListableBeanFactory beanFactor
        // Create a new XmlBeanDefinitionReader for the given BeanFactory.
        //创建xml的解析器，这里是一个委托模式
        XmlBeanDefinitionReader beanDefinitionReader = new XmlBeanDefinition

        // Configure the bean definition reader with this context's
        // resource loading environment.
        beanDefinitionReader.setEnvironment(this.getEnvironment());
        //这里传一个this进去，因为ApplicationContext是实现了ResourceLoader接口的
        beanDefinitionReader.setResourceLoader(this);
        beanDefinitionReader.setEntityRes

        // Allow a subclass to provide cu
        // then proceed with actually loa
        initBeanDefinitionReader(beanDefi
        //主要看这个方法   重要程度 5
        loadBeanDefinitions(beanDefinitio
    }
```

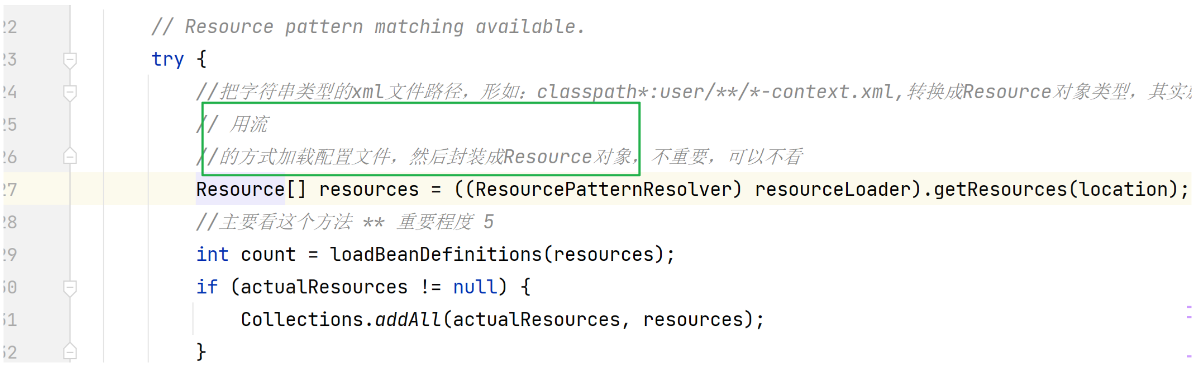
```
public void setResourceLoader(
    @Nullable org.springframework.core.io.ResourceLo
)
```

Set the ResourceLoader to use for resource locations. I
ResourcePatternResolver, the bean definition reader w
resource patterns to Resource arrays.
Default is PathMatchingResourcePatternResolver, also
pattern resolving through the ResourcePatternResolve
Setting this to null suggests that absolute resource lo
this bean definition reader.

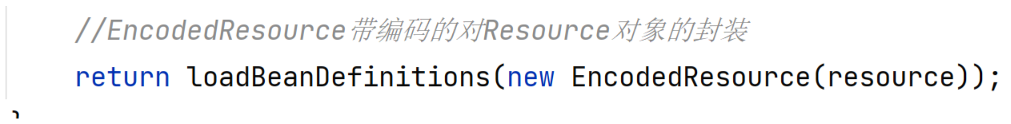
See Also: ResourcePatternResolver,
PathMatchingResourcePatternResolver

org.springframework.beans.factory.support.Abst

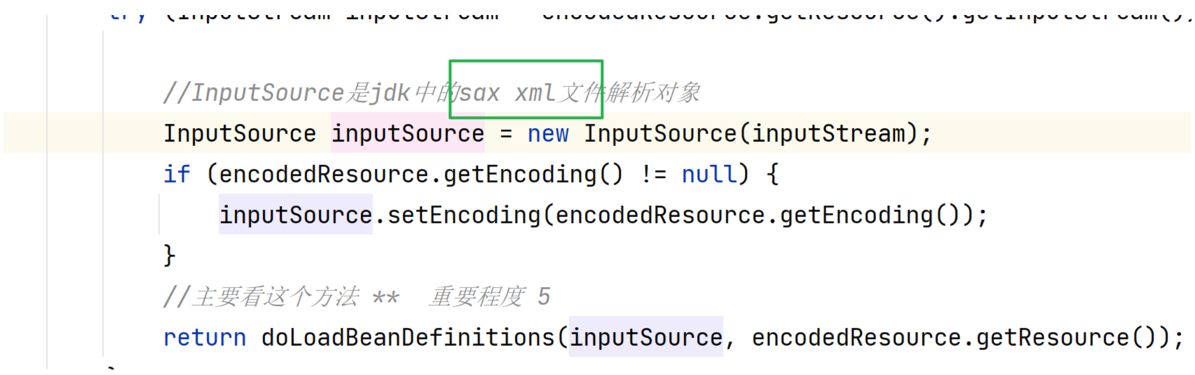
上下文接口实现资源加载接口



```java
        // Resource pattern matching available.
        try {
            //把字符串类型的xml文件路径，形如：classpath*:user/**/*-context.xml,转换成Resource对象类型，其实
            // 用流
            //的方式加载配置文件，然后封装成Resource对象，不重要，可以不看
            Resource[] resources = ((ResourcePatternResolver) resourceLoader).getResources(location);
            //主要看这个方法 ** 重要程度 5
            int count = loadBeanDefinitions(resources);
            if (actualResources != null) {
                Collections.addAll(actualResources, resources);
            }
```

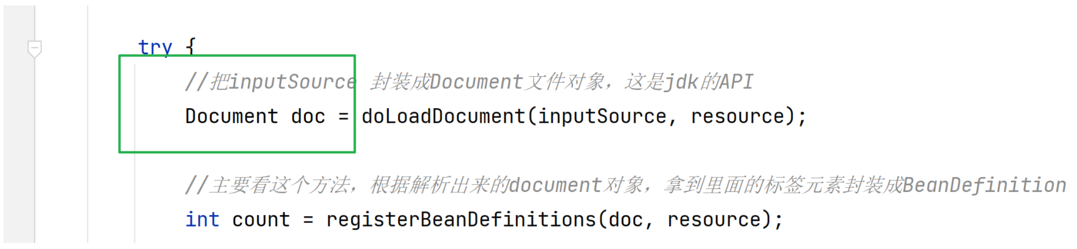


```java
    //EncodedResource带编码的对Resource对象的封装
    return loadBeanDefinitions(new EncodedResource(resource));
}
```



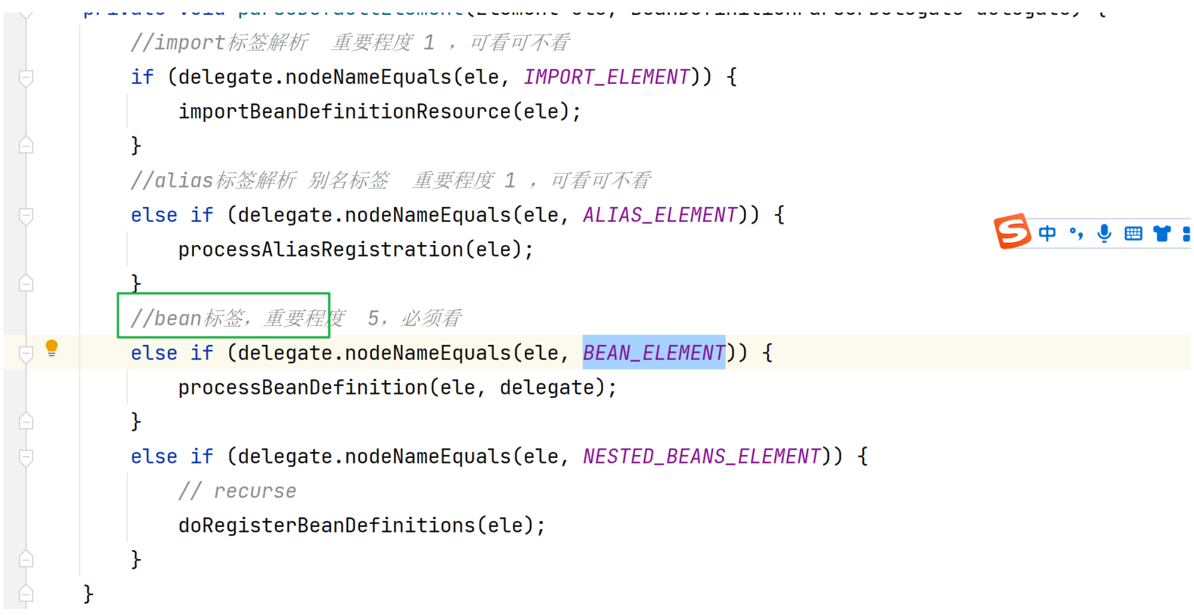
```java
    try (InputStream inputStream = encodedResource.getResource().getInputStream()) {

        //InputSource是jdk中的sax xml文件解析对象
        InputSource inputSource = new InputSource(inputStream);
        if (encodedResource.getEncoding() != null) {
            inputSource.setEncoding(encodedResource.getEncoding());
        }
        //主要看这个方法 **  重要程度 5
        return doLoadBeanDefinitions(inputSource, encodedResource.getResource());
```



```java
    try {
        //把inputSource 封装成Document文件对象，这是jdk的API
        Document doc = doLoadDocument(inputSource, resource);

        //主要看这个方法，根据解析出来的document对象，拿到里面的标签元素封装成BeanDefinition
        int count = registerBeanDefinitions(doc, resource);
```



```java
private void parseDefaultElement(Element ele, BeanDefinitionParserDelegate delegate) {
    //import标签解析  重要程度 1 ，可看可不看
    if (delegate.nodeNameEquals(ele, IMPORT_ELEMENT)) {
        importBeanDefinitionResource(ele);
    }
    //alias标签解析 别名标签  重要程度 1 ，可看可不看
    else if (delegate.nodeNameEquals(ele, ALIAS_ELEMENT)) {
        processAliasRegistration(ele);
    }
    //bean标签，重要程度  5，必须看
    else if (delegate.nodeNameEquals(ele, BEAN_ELEMENT)) {
        processBeanDefinition(ele, delegate);
    }
    else if (delegate.nodeNameEquals(ele, NESTED_BEANS_ELEMENT)) {
        // recurse
        doRegisterBeanDefinitions(ele);
    }
}
```

http://117.33.237.52:20070/desperado-image/GenericBeanDefinition.jpg