#### Spring Framework源码阅读

##### 之Application初始化：Spring Factories

问题描述

Spring Boot 2.2.5

Spring Framework 5.2.6

JDK 1.8

Spring Boot、Spring Framework IoC容器实现逻辑复杂，其中相当大一部分工作在解耦合。

Spring Factories机制就是Spring“解耦合”设计理念的一个落地实现。Spring Factories设计思路借鉴了Java SPI（Service Provider Interface）。

解决

Spring Boot应用初始化阶段，会加载Spring Factories，摘录关键代码如下：

private static Map<String, List<String>> loadSpringFactories(@Nullable ClassLoader classLoader) {  
 MultiValueMap<String, String> result = *cache*.get(classLoader);  
 if (result != null) {  
 return result;  
 }  
  
 try {  
 Enumeration<URL> urls = (classLoader != null ?  
 classLoader.getResources(*FACTORIES\_RESOURCE\_LOCATION*) :  
 ClassLoader.*getSystemResources*(*FACTORIES\_RESOURCE\_LOCATION*));  
 result = new LinkedMultiValueMap<>();  
 while (urls.hasMoreElements()) {  
 URL url = urls.nextElement();  
 UrlResource resource = new UrlResource(url);  
 Properties properties = PropertiesLoaderUtils.*loadProperties*(resource);  
 for (Map.Entry<?, ?> entry : properties.entrySet()) {  
 String factoryTypeName = ((String) entry.getKey()).trim();  
 for (String factoryImplementationName : StringUtils.*commaDelimitedListToStringArray*((String) entry.getValue())) {  
 result.add(factoryTypeName, factoryImplementationName.trim());  
 }  
 }  
 }  
 *cache*.put(classLoader, result);  
 return result;  
 }  
 catch (IOException ex) {  
 throw new IllegalArgumentException("Unable to load factories from location [" +  
 *FACTORIES\_RESOURCE\_LOCATION* + "]", ex);  
 }  
}

默认的加载路径为类路径"META-INF/spring.factories"。类似于ATG配置文件的加载方式，spring.factories配置文件采用叠加的方式进行加载。但是，由于采用MultiValueMap实现，同名工厂之间不会相互覆盖。

默认的spring.factories文件位于

org/springframework/boot/spring-boot/2.3.0.RELEASE/spring-boot-2.3.0.RELEASE.jar!/META-INF/spring.factories

□