原 Spring Boot的扩展机制之Spring Factories

2018年08月24日 13:56:28 伊成 阅读数 6862 更多

Spring Boot的扩展机制之Spring Factories

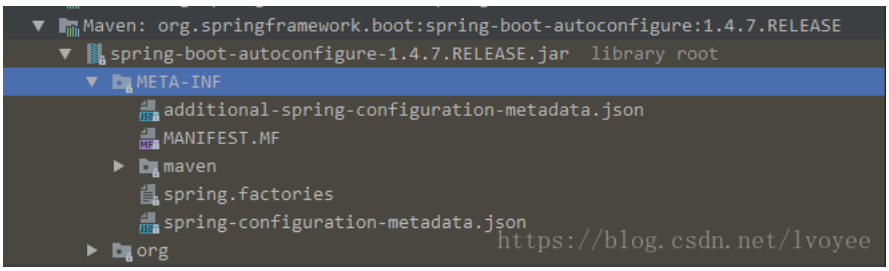
写在前面：Spring Boot中有一种非常解耦的扩展机制：Spring Factories。这种扩展机制实际上是仿照Java中的SPI扩展机制来实现的。

- 什么是 SPI机制

SPI的全名为Service Provider Interface.大多数开发人员可能不熟悉，因为这个是针对厂商或者插件的。在java.util.ServiceLoader的文档里有比较详细的
简单的总结下java SPI机制的思想。我们系统里抽象的各个模块，往往有很多不同的实现方案，比如日志模块的方案，xml解析模块、jdbc模块的方案等
象的设计里，我们一般推荐模块之间基于接口编程，模块之间不对实现类进行硬编码。一旦代码里涉及具体的实现类，就违反了可拔插的原则，如果需
现，就需要修改代码。为了实现在模块装配的时候能不在程序里动态指明，这就需要一种服务发现机制。
java SPI就是提供这样的一个机制：为某个接口寻找服务实现的机制。有点类似IOC的思想，就是将装配的控制权移到程序之外，在模块化设计中这个机
要。

- Spring Boot中的SPI机制

在Spring中也有一种类似与Java SPI的加载机制。它在META-INF/spring.factories文件中配置接口的实现类名称，然后在程序中读取这些配置文件并实例
这种自定义的SPI机制是Spring Boot Starter实现的基础。



- Spring Factories实现原理是什么

spring-core包里定义了SpringFactoriesLoader类，这个类实现了检索META-INF/spring.factories文件，并获取指定接口的配置的功能。在这个类中定义了
方法：

loadFactories 根据接口类获取其实现类的实例，这个方法返回的是对象列表。
loadFactoryNames 根据接口获取其接口类的名称，这个方法返回的是类名的列表。
上面的两个方法的关键都是从指定的ClassLoader中获取**spring.factories**文件，并解析得到类名列表，具体代码如下↓

```
public static List<String> loadFactoryNames(Class<?> factoryClass, ClassLoader classLoader) {
    String factoryClassName = factoryClass.getName();
    try {
        Enumeration<URL> urls = (classLoader != null ? classLoader.getResources(FACTORIES_RESOURCE_LOCATION) :
                ClassLoader.getSystemResources(FACTORIES_RESOURCE_LOCATION));
        List<String> result = new ArrayList<String>();
        while (urls.hasMoreElements()) {
            URL url = urls.nextElement();
            Properties properties = PropertiesLoaderUtils.loadProperties(new UrlResource(url));
            String factoryClassNames = properties.getProperty(factoryClassName);
            result.addAll(Arrays.asList(StringUtils.commaDelimitedListToStringArray(factoryClassNames)));
        }
        return result;
    }
    catch (IOException ex) {
        throw new IllegalArgumentException("Unable to load [" + factoryClass.getName() +
                "] factories from location [" + FACTORIES_RESOURCE_LOCATION + "]", ex);
    }
}
```

从代码中我们可以知道，在这个方法中会遍历整个ClassLoader中所有jar包下的spring.factories文件。也就是说我们可以在自己的jar中配置spring.factor
会影响到其它地方的配置，也不会被别人的配置覆盖。

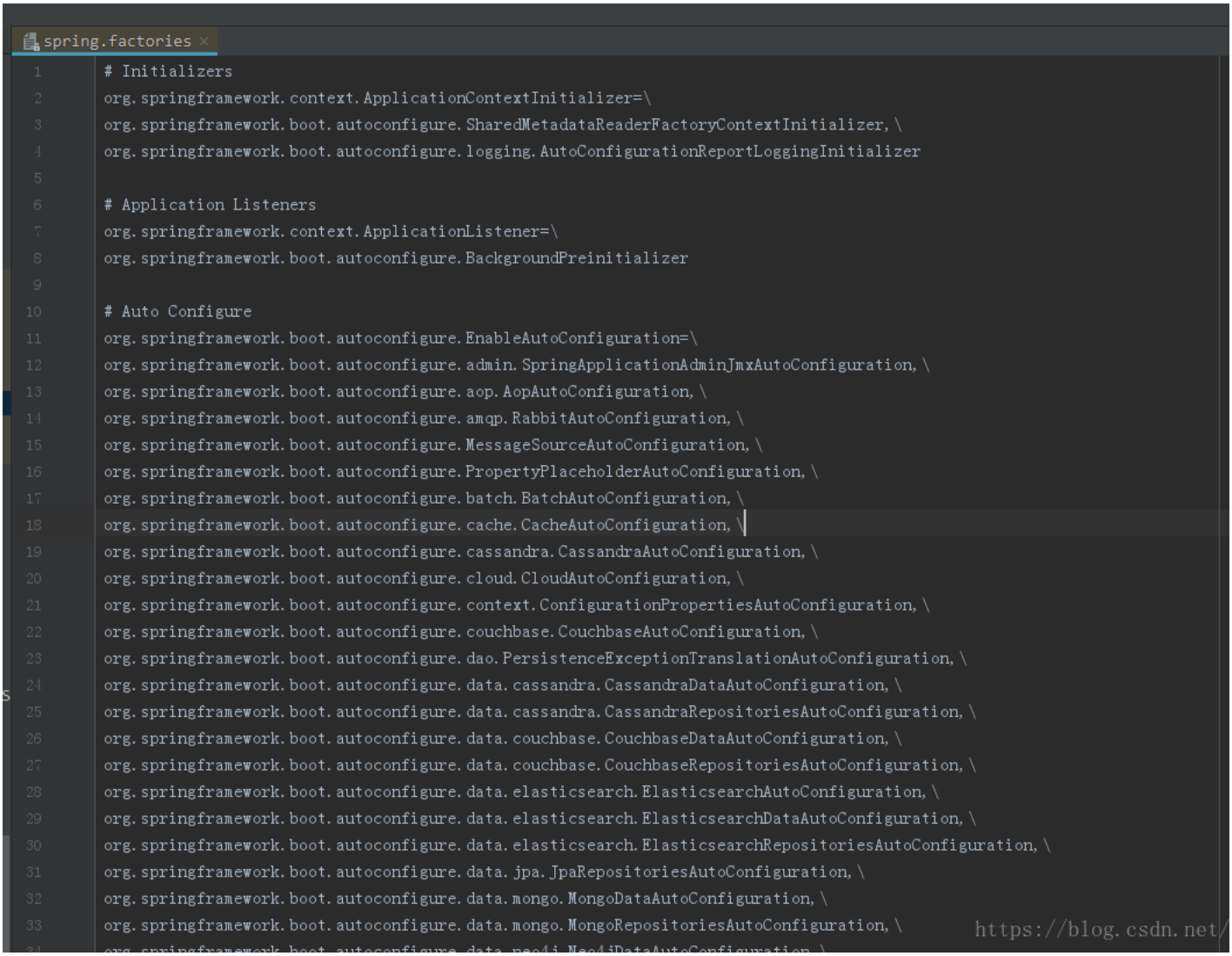
spring.factories的是通过Properties解析得到的，所以我们在写文件中的内容都是安装下面这种方式配置的：

com.xxx.interface=com.xxx.classname

如果一个接口希望配置多个实现类，可以使用','进行分割。

- Spring Factories在Spring Boot中的应用

在Spring Boot的很多包中都能够找到spring.factories文件，接下来我们以spring-boot包为例进行介绍



在日常工作中，我们可能需要实现一些SDK或者Spring Boot Starter给被人使用时，
我们就可以使用Factories机制。Factories机制可以让SDK或者Starter的使用只需要很少或者不需要进行配置，只需要在服务中引入我们的jar包即可。