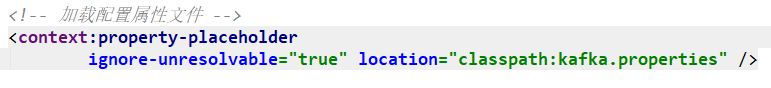
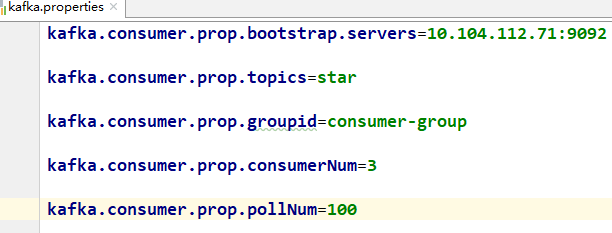
1. @value注解的使用

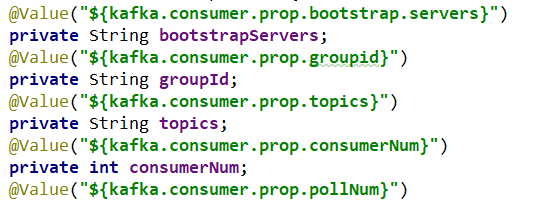
*<!-- 加载配置属性文件 -->*<**context:property-placeholder  
ignore-unresolvable="true" location="classpath:kafka.properties"** />



配置文件内容



注入注解



1. @PostConstruct注解的使用

被@PostConstruct修饰的方法会在服务器加载Servle的时候运行，并且只会被服务器执行一次。PostConstruct在构造函数之后执行,init()方法之前执行。

作用：

比如说我有一种情况，在我的servlet初始化加载之前我想处理一些东西，像加载缓存等等。

怎么做。@PostConstruct就派上用场了。那为什么这玩意用的不多呢，这是因为如果初始化之前我们要加载或处理某些玩意完全可以在构造器初始化时就处理了，但这种方法需要自己重写构造器。

1. 使用ApplicationContext获取同一类的所有实例，同时使用map代替if-else

@Component  
**public class** TopicConfig **implements** ApplicationContextAware {  
  
  
 **private** ApplicationContext **applicationContext**;  
  
 @Override  
 **public void** setApplicationContext(ApplicationContext applicationContext) **throws** BeansException {  
 **this**.**applicationContext** = applicationContext;  
 }  
  
 **public** Map<String, AbstractBaseLogFilter> getTopicType() {  
 Map<String,AbstractBaseLogFilter> map = **applicationContext**.getBeansOfType(AbstractBaseLogFilter.**class**);  
 Map<String,AbstractBaseLogFilter> result = **new** HashMap<>();  
 **for**(String str : map.keySet()){  
 result.put(str.substring(0,str.length()-16),map.get(str));  
 }  
 **return** result;  
 }  
}

