PropertyPlaceholderConfigurer

# PropertyPlaceholderConfigurer

org.springframework.beans.factory.config.PropertyPlaceholderConfigurer

PropertyPlaceholderConfigurer将**属性文件中的信息加载到XML配置文件中**，可以实现：在**单独的标准java Properties文件**中配置属性，然后在spring的xml配置文件中，通过**${key}**获取配置属性的效果，这样做可以达到**bean配置与properties配置**分离的目的。

PropertyPlaceholderConfigurer是个**bean工厂后置处理器的实现**，也就是 **BeanFactoryPostProcessor接口**的一个实现。

在Spring中，使用PropertyPlaceholderConfigurer可以在XML配置文件中加入外部属性文件，当然也可以指定外部文件的编码

.查看源代码，可以发现，locations属性定义在PropertyPlaceholderConfigurer的祖父类 PropertiesLoaderSupport中，而location只有 setter方法。类似于这样的配置，在spring的源程序中很常见的。

PropertyPlaceholderConfigurer如果在指定的Properties文件中找不到你想使用的属性，它还会在Java的System类属性中查找。

我们可以通过System.setProperty(key, value)或者java中通过-Dnamevalue来给Spring配置文件传递参数。

为简化PropertyPlaceholderConfigurer的使用，Spring提供了<context:property-placeholder/>元素。下面给出了配置示例，启用它后，开发者便不用配置PropertyPlaceholderConfigurer对象了。

<context:property-placeholder location="userinfo.properties"/>

　　PropertyPlaceholderConfigurer内置的功能非常丰富，如果它未找到${xxx}中定义的xxx键，它还会去JVM系统属性（System.getProperty()）和环境变量　　（System.getenv()）中寻找。通过启用systemPropertiesMode和searchSystemEnvironment属性，开发者能够控制这一行为。

这个在spring中配置文件中是非常常用的。

context:property-placeholder大大的方便了我们数据库的配置。

只需要在spring的配置文件里添加一句：**<context:property-placeholder?location="classpath:jdbc.properties"/>**?即可，这里location值为参数配置文件的位置.

## 使用示例

### 在resources/properties目录下创建几个properties文件：

db.properties、redis.properties、jsf.properties等

### 在xml中配置引入properties文件

<!-- 读取属性配置文件 -->

<bean id="**propertyConfigurer**" class="org.springframework.beans.factory.config.**PropertyPlaceholderConfigurer**">

<!--多properties文件配置-->

<property name="**locations**">

<list>

<value>classpath:db.properties</value>

<value>classpath:r2m.properties</value>

<value>classpath:auth.properties</value>

<value>classpath:quartz.properties</value>

<value>classpath:email.properties</value>

<value>classpath:jsf.properties</value>

</list>

</property>

<property name="fileEncoding" value="UTF-8"/>

<property name="ignoreUnresolvablePlaceholders" value="true" />

</bean>

### 在xml文件中利用${propertyKey}即可将value引入。



# 问题：





单独运行A模块，或单独运行B模块都是正常的，但将A和B两个模块集成后运行，Spring容器就启动不了了：

Could not resolve placeholder 'moduleb.jdbc.driverClassName' in string value "${moduleb.jdbc.driverClassName}"

原因：**Spring容器采用反射扫描的发现机制**，在探测到Spring容器中有一个org.springframework.beans.factory.config.PropertyPlaceholderConfigurer的Bean就会停止对剩余PropertyPlaceholderConfigurer的扫描

spring容器中最多只能定义一个context:property-placeholder，不然就出现那种个错误，那如何来解决上面的问题呢？



1. <context:property-placeholder location="classpath\*:conf/conf\*.properties"/>
2. <import resource="a.xml"/>
3. <import resource="b.xml"/>

<https://www.cnblogs.com/huqianliang/p/5673701.html>