# context:property-placehoder

这个在spring中配置文件中是非常常用的。

只需要在spring的配置文件里添加一句：

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

location：值为参数配置文件的位置，参数配置文件通常放在src目录下，而参数配置文件的格式跟java通用的参数配置文件相同，即键值对的形式。

如：db.properties

db.driverclass=oracle.jdbc.driver.OracleDriver

db.url=jdbc:oracle:thin:@localhost:1521:JEDA

db.username=scott

db.password=zzs

这样一来，就可以为spring配置的bean的属性设置值了，比如spring有一个jdbc数据源的类

<!-- 配置数据源 -->

<!-- 导入资源文件 -->

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

<bean id="dataSource" class="org.apache.tomcat.dbcp.dbcp.BasicDataSource">

<property name="driverClassName" value="${db.driverclass}"></property>

<property name="url" value="${db.url}"></property>

<property name="username" value="${db.username}"></property>

<property name="password" value="${db.password}"></property>

</bean>

这样修改起来比较方便。

如果多个需要加载，可以这样写

<context:property-placeholder location=”classpath:db\*.properties”/>

<import resource=”a.xml”/>

<import resource=”b.xml”/>

# context:component-scan

@Controller 声明Action组件

@Service 声明Service组件 @Service("myMovieLister")

@Repository 声明Dao组件

@Component 泛指组件, 当不好归类时.

@RequestMapping("/menu") 请求映射

@Resource 用于注入，( j2ee提供的 ) 默认按名称装配，@Resource(name="beanName")

@Autowired 用于注入，(srping提供的) 默认按类型装配

@Transactional( rollbackFor={Exception.class}) 事务管理

@ResponseBody

@Scope("prototype") 设定bean的作用域

在xml配置了这个标签后，spring可以自动去扫描base-pack下面或者子包下面的java文件，如果扫描到有@Component @Controller @Service @Repository等这些注解的类，则把这些类注册为bean

注意：如果配置了<context:component-scan>那么<context:annotation-config/>标签就可以不用再xml中配置了，因为前者包含了后者。另外<context:annotation-config/>还提供了两个子标签

1. <context:include-filter>

2. <context:exclude-filter>

在说明这两个子标签前，先说一下<context:component-scan>有一个use-default-filters属性，该属性默认为true,这就意味着会扫描指定包下的全部的标有@Component的类，并注册成bean.也就是@Component的子注解@Service,@Reposity等。所以如果仅仅是在配置文件中这么写

<context:component-scan base-package="tv.huan.weisp.web"/>

Use-default-filter此时为true那么会对base-package包或者子包下的所有的进行java类进行扫描,并把匹配的java类注册成bean。

可以发现这种扫描的粒度有点太大，如果你只想扫描指定包下面的Controller，该怎么办？此时子标签<context:incluce-filter>就起到了勇武之地。如下所示

<context:component-scan base-package="tv.huan.weisp.web .controller">

<context:include-filter type="annotation" expression="org.springframework.stereotype. Controller"/>

</context:component-scan>

这样就会只扫描base-package指定下的有@Controller下的java类，并注册成bean

但是因为use-dafault-filter在上面并没有指定，默认就为true，所以当把上面的配置改成如下所示的时候，就会产生与你期望相悖的结果（注意base-package包值得变化）

<context:component-scan base-package="tv.huan.weisp.web ">

<context:include-filter type="annotation" expression="org.springframework.stereotype. Controller"/>

</context:component-scan>

此时，spring不仅扫描了@Controller，还扫描了指定包所在的子包service包下注解@Service的java类

此时指定的include-filter没有起到作用，只要把use-default-filter设置成false就可以了。这样就可以避免在base-packeage配置多个包名这种不是很优雅的方法来解决这个问题了。

另外在我参与的项目中可以发现在base-package指定的包中有的子包是不含有注解了，所以不用扫描，此时可以指定<context:exclude-filter>来进行过滤，说明此包不需要被扫描。综合以上说明

Use-dafault-filters=”false”的情况下：<context:exclude-filter>指定的不扫描，<context:include-filter>指定的扫描

# <context:annotation-config />

<context:annotation-config />标签意义：

该标签隐式的向Spring容器注册了：

AutowiredAnnotationBeanPostProcessor CommondAnnotationBeanPostProcessor

PersistenceAnnotationBeanPostProcessor RequiredAnnotationBeanPostProcessor这四个BeanPostProcessor.

1.AutowiredAnnotationBeanPostProcessor:主要是使用@AutoWired注解时，必须提前向Spring容器注册这个BeanPostprocessor

传统的配置方式：<bean class="org.springframework.beans.factory.annotation. AutowiredAnnotationBeanPostProcessor "/>

2.CommondAnnotationBeanPostprocessor:主要是使用@Resource、@PostConstruct、@Predestory等注解时，必须提前向Spring容器注册。

传统配置方式：<bean class="org.springframework.beans.factory.annotation.CommondAnnotationBeanPostprocessor"/>

3.PersistenceAnnotationBeanPostProcessor :主要是使用@PersistenceContext注解时，必须提前向Spring容器注册。

传统配置方式：<bean class="org.springframework.beans.factory.annotation.PersistenceAnnotationBeanPostProcessor"/>

4.RequiredAnnotationBeanPostProcessor:主要是使用@Required注解时，必须提前向Spring容器注册。

传统配置方式：<bean class="org.springframework.beans.factory.annotation.RequiredAnnotationBeanPostProcessor"/>

一般来说，@Autowired注解比较常用，传统配置方式有点繁琐，而这种隐式配置方式就可以帮我们很容易的搞定。

不过使用注解，我们一般都会配置扫描包路径：<context: component-scan base-package=com.xxx.xxx />

其实，给配置已经包含了自动注入上述Processor的功能，所以，配置了包扫描之后，<context annotation-config />就可以移除了。

# <mvc:annotation-driven />

<mvc:annotation-driven /> 是一种简写形式，完全可以手动配置替代这种简写形式，简写形式可以让初学都快速应用默认配置方案。<mvc:annotation-driven /> 会自动注册DefaultAnnotationHandlerMapping与AnnotationMethodHandlerAdapter 两个bean,是spring MVC为@Controllers分发请求所必须的。

并提供了：数据绑定支持，@NumberFormatannotation支持，@DateTimeFormat支持，@Valid支持，读写XML的支持（JAXB），读写JSON的支持（Jackson）。

后面，我们处理响应ajax请求时，就使用到了对json的支持。

后面，对action写JUnit单元测试时，要从spring IOC容器中取DefaultAnnotationHandlerMapping与AnnotationMethodHandlerAdapter 两个bean，来完成测试，取的时候要知道是<mvc:annotation-driven />这一句注册的这两个bean。

<context:annotation-config> declares support for general annotations such as @Required, @Autowired, @PostConstruct, and so on.

<mvc:annotation-driven /> is actually rather pointless. It declares explicit support for annotation-driven MVC controllers (i.e.@RequestMapping, @Controller, etc), even though support for those is the default behaviour.

My advice is to always declare <context:annotation-config>, but don't bother with <mvc:annotation-driven /> unless you want JSON support via Jackson.