**使用路径通配符加载Resource,**

原创 2013年01月11日 11:04:28

* 8921

Spring提供了强大的Ant模式通配符匹配，从同一个路径能匹配一批资源。

Ant路径通配符支持"?"、"\*"、"\*\*"，注意通配符匹配不包括目录分隔符“/”。

“?”:匹配一个字符，如"config?.xml"可匹配"config1.xml".

 “\*”：匹配零个或多个字符串，如“com/\*/config.xml”将匹配“cn/feng/config.xml”，但不匹配匹配“com/config.xml”(因为这里匹配的是字符串，如果是目录的话则可以)；而“com/config-\*.xml”将匹配“com/config-dao.xml”；

"\*\*"：匹配路径中的零个或多个目录。如“com/\*\*/config.xml”将匹配“com/config.xml”，也匹配“com/feng/spring/config.xml”；而“com/feng/config-\*\*.xml”将匹配“com/feng/config-dao.xml”，即把“\*\*”当做两个“\*”处理。

Spring在加载类路径资源时除了提供前缀“classpath:”的来支持加载一个Resource，还提供一个前缀“classpath\*:”来支持加载所有匹配的类路径Resource。

Spring提供了ResourcePatternResolver接口来加载多个Resource.

**[java]** [view plain](http://blog.csdn.net/howlaa/article/details/8492289) [copy](http://blog.csdn.net/howlaa/article/details/8492289)

1. **package** com.feng.spring.chapter2.helloworld;
3. **import** java.io.IOException;
5. **import** org.springframework.core.io.Resource;
6. **import** org.springframework.core.io.ResourceLoader;
8. **public** **interface** ResourcePatternResolver **extends** ResourceLoader{
9. String CLASSPATH\_ALL\_URL\_PREFIX = "classpath\*:";
10. Resource[] getResources(String locationPattern) **throws** IOException;//添加了此方法用来接收多个Resource
11. }

一、"classpath":用于加载类路径(包括jar包)中的一个且仅一个资源;对于多个匹配的也只返回一个。如果需要多个匹配的则考虑"classpath\*."前缀。

**[java]** [view plain](http://blog.csdn.net/howlaa/article/details/8492289) [copy](http://blog.csdn.net/howlaa/article/details/8492289)

1. **package** com.feng.spring.chapter2.helloworld;

4. **import** java.io.IOException;
6. **import** org.junit.Test;
7. **import** org.springframework.core.io.Resource;
8. **import** org.springframework.core.io.support.PathMatchingResourcePatternResolver;

11. **public** **class** HelloTest {
12. @Test
13. **public** **void** testClasspathPrefix()**throws** IOException{
14. ResourcePatternResolver resolver = (ResourcePatternResolver) **new** PathMatchingResourcePatternResolver();
15. //只加载一个绝对匹配Resource,且通过ResourceLoader.getResource进行加载
16. Resource resources=resolver.getResource("classpath:META-INF/INDEX.LIST");
17. Assert.assertEquals(1, resources.length);
18. //只加载一个匹配的Resource,且通过ResourceLoader.getResource进行加载
19. resources = resolver.getResource("classpath:META-INF/\*.LIST");
20. Assert.assertTrue(resources.length == 1);
21. }
22. }

二、"classpath\*":

用于加载类路径(包括jar包)中所有的匹配的资源。

**[java]** [view plain](http://blog.csdn.net/howlaa/article/details/8492289) [copy](http://blog.csdn.net/howlaa/article/details/8492289)

1. **package** com.feng.spring.chapter2.helloworld;

4. **import** java.io.IOException;
6. **import** javax.annotation.Resource;
8. **import** junit.framework.Assert;
10. **import** org.junit.Test;
11. **import** org.springframework.core.io.support.PathMatchingResourcePatternResolver;

14. **public** **class** HelloTest {
15. @Test
16. **public** **void** testClasspathAsteriskPrefix()**throws** IOException{
17. ResourcePatternResolver resolver = (ResourcePatternResolver) **new** PathMatchingResourcePatternResolver();
19. //将加载多个绝对匹配的所有Resource
20. //将首先通过ClassLoader.getResource("META-INF")加载非模式路径部分
21. //然后进行遍历模式匹配
22. Resource[] resources = (Resource[]) resolver.getResources("classpath\*:META-INF/INDEX.LIST");
23. Assert.assertTrue(resources.length > 1);
24. //将加载多个模式匹配的Resource
25. resources = (Resource[]) resolver.getResources("classpath\*:META-INF/\*.LIST");
26. Assert.assertTrue(resources.length > 1);
27. }
28. }

带通配符的classpath使用“ClassLoader”的“Enumeration<URL> getResources(String name)”方法来查找通配符之前的资源，然后通过模式匹配来获取匹配的资源。如“classpath:META-INF/\*.LIST”将首先加载通配符之前的目录“META-INF”，然后再遍历路径进行子路径匹配从而获取匹配的资源。

三、"file":加载一个或多个系统中的Resource。如:"file:D/\*.txt"将返回D盘下的所有txt文件。

四、无前缀：通过ResourceLoader实现加载一个资源。

ApplicationContext提供的getResource方法将获取资源委托给ResourcePatternResolver实现，默认使用PathMatingResourcePatternResolver.