链滴

# Spring 高级装配

作者：jiangyue

原文链接：https://ld246.com/article/1478783662477

来源网站：链滴

 
<ul>
<li>
在 JavaConfig 中使用 <code>@Profile</code> 注解指定某个 bean 属于哪个 profile，只有规定 profile时 bean 才会被创建，没有指定 profile 的 bean 会始终被创建。
</li>
<li>
在 xml 中可以通过 <code><beans></code> 元素的 profile 属性配置 bean，还可以在<code><beans></code>元素中嵌套定义<code><beans></code>元素，而不是为一个 profile 创建一个 xml 配置。
</li>
<li>
spring 通过 spring.profiles.active 和 spring.profiles.default</trong> 确定激活哪一个 profile，如果设置了 active，将用 active 的值作为激活的 profile，否则将使用 default 的值作为激活的 profile。如果均未设置，表示没有激活的 profile，只创建没有定义在 pofile 中的 bean。
</li>
<li>
有多种方式可以设置配置 profile 的两个属性(active 和 default): 作为 DispatcherServlet 的初始化参数，作为 web.xml 参数，作为 JDNI 条目，作为环境变量，<strog>作为 JVM 系统属性，在集成测试类上使用<code>@ActiveProfiles</code 注解设置。
</li>
<li>
可以在 web.xml 中使用<code><context-param></code>为上下文设置默认的 profile 在生产环境中根据情况设置 spring.profiles.active，方便开发人员从版本控制件中获取源码。
</li>
<li>
<code>@ActiveProfiles</code> 注解可以指定在运行测试时使用哪个 profile。
</li>
<li>
<code>@Conditional(yourCondition.class)</code> 注解可以用在带有<code>@Bean</coe>注解的方法上，给定条件满足时才会创建 bean，yourCondition.class 需要现 Condition 接口。
</li>
<li>
可以通过 <code>@Primary</code> 注解设置首选 bean 来解决自动装配中多个 bean 满足件的歧义性，在 xml中可以通过<code><bean primary="true" /></code>来设置。
</li>
<li>
装配限定符 <code>@Qualifier("qualifierName")</code> 可以与 <code>@Autowired</coe> 协同使用，表明要注入哪个 bean。<code>@Qualifier("qualifierName")</code> 也表示给 ben 指定限定符，没有指定 bean 的限定符时，会有一个默认的限定符与 ID 相同。<code>@Qualifier/code> 的最佳实践是为 bean 选择特征性或描述性的术语。需要同时使用多个限定符时，可以创建定义的限定符注解。
</li>
<li>
默认情况下，spring 应用上下文中 bean 都以单例 Singleton的形式创建除单例外，spring 还提供多种作用域：原型 Prototype，会话 Sessin，请求 Request。
</li>
<li>

选择其他作用域，要使用 <code>@Scope</code> 注解，例如<code>@Scope(ConfigurablBeanFactory.SCOPE_PROTOTYPE)</code>。xml 中可以使用 bean 下的 scope 属性来设置。</p

</li>
<li>
<code>@Scope</code> 注解还有一个proxyMode属性，用来指定将他作用域的 bean 装配到单例 bean 中时的代理模式，<code>ScopedProxyMode.INTERFACES</code>表明要实现对应接口，<code>ScopedProxyMode.TARGET_CLASS</code>表明要以生成目扩展类的方式创建代理。
</li>
<li>
spring 处理外部值时，可以声明属性源并通过 spring 的 Environment 来检索属性，例如使用注解 <code>@PropertySource("classpath:/com/xxx/aaa.roperties")</code>，注入 Environment env，通过 env 的 <code>getProprty()</code>方法获取。如果要求属性必须定义，可以使用 <code>env.getRequiredProperty("xxx"</code>获取，在没有找到定义时抛出异常。
</li>
<li>
spring支持将属性定义到外部的属性的文件中，通过占位符的形式为使用 <code>${...}</code>的属性名称包装。
</li>
<li>
spring 的表达式语言 SpEL 要放在 <code>#{...}</code> 中，<code>T(</code>运算符的结果为 class 对象，能够使用目标类型的静态方法和常亮。
</li>
<li>
SpEL 通过 <code>matches</code> 运算符支持表达式模式匹配，返回 boolean 值，例如 <code>#{admin.email matches '[a-zA-Z0-9._%+-]+@[a-zA-Z0-9._]+\\.com'}</code> 匹配email址。
</li>
</ul>
<style><!--
h1,
h2,
h3,
h4,
h5,
h6,
p,
blockquote {
    margin: 0;
    padding: 0;
}
body {
    font-family: "Helvetica Neue", Helvetica, "Hiragino Sans GB", Arial, sans-serif;
    font-size: 13px;
    line-height: 18px;
    color: #737373;
    background-color: white;
    margin: 10px 13px 10px 13px;
}
table {
    margin: 10px 0 15px 0;

```
    border-collapse: collapse;
}
td,th {
    border: 1px solid #ddd;
    padding: 3px 10px;
}
th {
    padding: 5px 10px;
}

a {
color: #0069d6;
}
a:hover {
color: #0050a3;
text-decoration: none;
}
a img {
border: none;
}
p {
margin-bottom: 9px;
}
h1,
h2,
h3,
h4,
h5,
h6 {
color: #404040;
line-height: 36px;
}
h1 {
margin-bottom: 18px;
font-size: 30px;
}
h2 {
font-size: 24px;
}
h3 {
```

```
font-size: 18px;
}
h4 {
font-size: 16px;
}
h5 {
font-size: 14px;
}
h6 {
font-size: 13px;
}
hr {
margin: 0 0 19px;
border: 0;
border-bottom: 1px solid #ccc;
}
blockquote {
padding: 13px 13px 21px 15px;
margin-bottom: 18px;
font-family:georgia,serif;
font-style: italic;
}
blockquote:before {
content:"\201C";
font-size:40px;
margin-left:-10px;
font-family:georgia,serif;
color:#eee;
}
blockquote p {
font-size: 14px;
font-weight: 300;
line-height: 18px;
margin-bottom: 0;
font-style: italic;
}
code, pre {
```

```
font-family: Monaco, Andale Mono, Courier New, monospace;
}
code {
background-color: #fee9cc;
color: rgba(0, 0, 0, 0.75);
padding: 1px 3px;
font-size: 12px;
-webkit-border-radius: 3px;
-moz-border-radius: 3px;
border-radius: 3px;
}
pre {
display: block;
padding: 14px;
margin: 0 0 18px;
line-height: 16px;
font-size: 11px;
border: 1px solid #d9d9d9;
white-space: pre-wrap;
word-wrap: break-word;
}
pre code {
background-color: #fff;
color:#737373;
font-size: 11px;
padding: 0;
}
sup {
font-size: 0.83em;
vertical-align: super;
line-height: 0;
}
● {
-webkit-print-color-adjust: exact;
}
@media screen and (min-width: 914px) {
body {
```

```
width: 854px;
margin:10px auto;
}
}
@media print {
body,code,pre code,h1,h2,h3,h4,h5,h6 {
color: black;
}
table, pre {
page-break-inside: avoid;
}
}
--></style>
```