https://blog.csdn.net/u010002184/article/details/79353696

**ConditionalOnProperty的使用**

2018年02月23日 15:02:26 [二十六画生的博客](https://me.csdn.net/u010002184) 阅读数 21094

 版权声明：本文为博主原创文章，转载请说明出处 https://blog.csdn.net/u010002184/article/details/79353696

|  |
| --- |
| @Retention(RetentionPolicy.***RUNTIME***)  @Target({ElementType.***TYPE***, ElementType.***METHOD***})  @Documented  @Conditional({OnPropertyCondition.**class**})  **public** **@interface** ConditionalOnProperty {  String[] value() **default** {}; //数组，获取对应property名称的值，与name不可同时使用    String prefix() **default** "";//property名称的前缀，可有可无    String[] name() **default** {};//数组，property完整名称或部分名称（可与prefix组合使用，组成完整的property名称），与value不可同时使用    String havingValue() **default** "";//可与name组合使用，比较获取到的属性值与havingValue给定的值是否相同，相同才加载配置    **boolean** matchIfMissing() **default** **false**;//缺少该property时是否可以加载。如果为true，没有该property也会正常加载；反之报错    **boolean** relaxedNames() **default** **true**;//是否可以松散匹配，至今不知道怎么使用的  } |

|  |
| --- |
| @Configuration  @ConditionalOnProperty(value = "parentName.sonName")  public class parentNameConfig {  }  .yml配置如下：  parentName:  sonName: true //正常  parentName:  sonName: //正常，空字符时  parentName:  sonName: false //失败  parentName:  sonName: null //正常  parentName:  sonName: 123 //正常    把第二行换成：  @ConditionalOnProperty(value = "parentName.sonName",havingValue = "123")  .yml配置如下：  parentName:  sonName: 123 //正常  parentName:  sonName: 1234 //失败  parentName:  sonName: false //失败    把第二行换成：  @ConditionalOnProperty(value = "parentName.sonName",havingValue = "false")  .yml配置如下：  parentName:  sonName: false //正常    把第二行换成：  @ConditionalOnProperty(prefix = "parentName",name = "sonName",havingValue = "123")  .yml配置如下：  parentName:  sonName: 123 //正常  parentName:  sonName: 1234 //失败，与havingValue给定的值不一致    把第二行换成：  @ConditionalOnProperty(prefix = "parentName",name = "sonName",havingValue = "123",matchIfMissing = true)  .yml配置如下：  不配置相关参数 //正常，当matchIfMissing = true时，即使没有该parentName.sonName属性也会加载正常      把第二行换成：  @ConditionalOnProperty(prefix = "parentName",name = "sonName",havingValue = "123",matchIfMissing = false) //这里matchIfMissing默认为false，写不写都行  .yml配置如下：  不配置相关参数 //失败，当matchIfMissing = false时，必须要有对应的property  parentName:  sonName: 123 //正常    把第二行换成：  @ConditionalOnProperty(prefix = "parentName.", name = "sonName")//prefix带了.（点）  .yml配置如下：  parentName:  sonName: true //正常  parentName:  sonName: 123 //正常    把第二行换成：  @ConditionalOnProperty(prefix = "parentName",value = {"sonName2"},name = {"sonName"})  .yml配置如下：  parentName:  sonName: true //项目Debug启动失败，The name and value attributes of @ConditionalOnProperty are exclusive  //@ConditionalOnProperty的name和value属性是互斥的，不能同时出现    把第二行换成：  @ConditionalOnProperty(prefix = "parentName",name = {"sonName"})  .yml配置如下：  parentName:  sonName: true //正常    把第二行换成：  @ConditionalOnProperty(prefix = "parentName",name = {"sonName","flag"}) //name中的属性需要两个都存在且都不为false才会加载正常  .yml配置如下：  parentName:  sonName: true  flag: true //正常  parentName:  sonName: true  flag: 123 //正常  parentName:  sonName: true  flag: false //失败  parentName:  sonName: false  flag: false //失败    把第二行换成：  @ConditionalOnProperty(prefix = "parentName", name = {"sonName", "flag"}, havingValue = "false")  parentName:  sonName: false  flag: false //正常    把第二行换成：  @ConditionalOnProperty(prefix = "parentName", name = {"sonName", "flag"}, havingValue = "123")//parentName.sonName和parentName.flag的值都要与havingValue的一致才行  parentName:  sonName: 123  flag: 1234 //失败  parentName:  sonName: 123  flag: 123 //正常  parentName:  sonName: 123 //失败，缺少parentName.flag    把第二行换成：  @ConditionalOnProperty(prefix = "parentName", name = {"sonName", "flag"}, havingValue = "123",matchIfMissing = true)//matchIfMissing = true允许缺少  parentName:  sonName: 123 //正常  .yml配置如下：  不配置相关参数 //正常    把第二行换成：  @ConditionalOnProperty(prefix = "parentName", name = {"sonName", "flag"}, havingValue = "123")  .yml配置如下：  不配置相关参数 //失败 |