thymeleaf数据字典方言

## 1、配置文件类加载生成Bean PubCodeDialect

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | /\*\* | |  | \* 在Thymeleaf的自动配置前先将附加的Thymeleaf自定义方言注入进去 | |  | \*/ | |  | @Configuration | |  | @AutoConfigureBefore(ThymeleafAutoConfiguration.class) | |  | public class ThymeleafAdditionalDialect {  @ConditionalOnMissingBean注解判断是否执行初始化代码，即如果用户已经创建了bean，则相关的初始化代码不再执行 | |  | @Bean | |  | @ConditionalOnMissingBean | |  | public PubCodeDialect pubcodeDialect() { | |  | return new PubCodeDialect(); | |  | } | |  |  | |  | @Bean | |  | @ConditionalOnMissingBean | |  | public ShiroDialect shiroDialect() { | |  | return new ShiroDialect(); | |  | } | |  | } | |

## 2、PubCodeDialect 编写方言，继承抽象方言AbstractDialect，实现表达的对象方言IExpressionObjectDialect，主要是用来返回一个扩展的PubCodeExpressionFactory

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | /\*\* | |  | \* 公共代码表达式处理的Thymeleaf方言 | |  | \*/ | |  | public class PubCodeDialect extends AbstractDialect implements IExpressionObjectDialect { | |  | public static final String NAME = "PubCode Dialect"; | |  | private final IExpressionObjectFactory PUBCODE\_EXPRESSION\_OBJECT\_FACTORY = new PubCodeExpressionFactory(); | |  |  | |  | public PubCodeDialect(){ | |  | super(NAME); | |  | } | |  |  | |  | @Override | |  | public IExpressionObjectFactory getExpressionObjectFactory(){ | |  | return PUBCODE\_EXPRESSION\_OBJECT\_FACTORY; | |  | } | |  | } | |

## 3、PubCodeExpressionFactory 类

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | /\*\* | |  | \* 公共代码表达式创建的工厂类 | |  | \*/ | |  | public class PubCodeExpressionFactory implements IExpressionObjectFactory { | |  | private static final String PUBCODE\_VARIABLE\_NAME = "pubcode"; | |  | private static final Set<String> ALL\_EXPRESSION\_OBJECT\_NAMES = Collections.unmodifiableSet( | |  | new LinkedHashSet<>(Arrays.asList(PUBCODE\_VARIABLE\_NAME))); | |  |  | |  | @Override | |  | public Set<String> getAllExpressionObjectNames(){ | |  | return ALL\_EXPRESSION\_OBJECT\_NAMES; | |  | } | |  |  | |  | @Override | |  | public Object buildObject(IExpressionContext context, final String expressionObjectName) { | |  | if (PUBCODE\_VARIABLE\_NAME.equalsIgnoreCase(expressionObjectName)) { | |  | return new PubCodeExpression(); | |  | } | |  | return null; | |  | } | |  |  | |  | @Override | |  | public boolean isCacheable(String expressionObjectName) { | |  | return expressionObjectName != null && PUBCODE\_VARIABLE\_NAME.equalsIgnoreCase(expressionObjectName); | |  | } | |  | } | |

## 4、从工厂中取出

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | /\*\* | |  | \* 公共代码处理的表达式 | |  | \*/ | |  | public final class PubCodeExpression { | |  | public PubCodeExpression(){} | |  |  | |  | /\*\* | |  | \* 根据类型和代码获取该公共代码的值 | |  | \* @param type 类型 | |  | \* @param code 代码 | |  | \* @return | |  | \*/ | |  | public String codeName(String type, String code) { | |  | return PubCodeUtil.getName(type, code); | |  | } | |  | } | |

## 5、前段thyme开始使用