**Thymeleaf 的基本语法**

**Thymeleaf是Web和独立环境的现代服务器端Java模板引擎，能够处理HTML，XML，JavaScript，CSS甚至纯文本。**

**Thymeleaf的主要目标是提供一种优雅和高度可维护的创建模板的方式。为了实现这一点，它建立在自然模板的概念上，将其逻辑注入到模板文件中，不会影响模板被用作设计原型。这改善了设计的沟通，弥补了设计和开发团队之间的差距。**

**Thymeleaf也从一开始就设计了Web标准 - 特别是HTML5 - 允许您创建完全验证的模板，如果这是您需要的**

**springboot 用thymeleaf 还是挺不错的**

温馨提示： 点击右边 展示皮肤 –> 选择 经典白 这个主题可能会更加适合。

**一、标准表达式语法**

**它又分为：**

* 消息
* 变量
* 选择表达式
* 链接URL
* 片段
* 文字
* 附加文本
* 字面替代
* 算术运算
* 比较与平等
* 条件表达式
* 默认表达式
* 无操作令牌
* 数据转换/格式化
* 预处理

**我就只介绍常用的了**

${…} 表达式实际上是在上下文中包含的变量的地图上执行的OGNL（Object-Graph Navigation Language）对象。

**1、变量**

<p>Today is: <span th:text="${today}">13 february 2011</span>.</p>

意味着 <span> 标签中的内容会被表达式${today}的值所替代，无论模板中它的内容是什么，之所以在模板中“多此一举“地填充它的内容，完全是为了它能够作为原型在浏览器中直接显示出来。   
假设today的值为2015年8月14日，那么渲染结果为：<p>Today is: 2015年8月14日.</p>。可见Thymeleaf的基本变量和JSP一样，都使用${.}表示获取变量的值。

**2、URL**

**URL在Web应用模板中占据着十分重要的地位，需要特别注意的是Thymeleaf对于URL的处理是通过语法@{…}来处理的。Thymeleaf支持绝对路径URL：**

<a th:href="@{http://www.thymeleaf.org}">Thymeleaf</a>

**同时也能够支持相对路径URL：**

**另外，如果需要Thymeleaf对URL进行渲染，那么务必使用th:href，th:src等属性，下面是一个例子**

<!-- Will produce 'http://localhost:8080/gtvg/order/details?orderId=3' (plus rewriting) -->

<a href="details.html"

th:href="@{http://localhost:8080/gtvg/order/details(orderId=${o.id})}">view</a>

<!-- Will produce '/gtvg/order/details?orderId=3' (plus rewriting) -->

<a href="details.html" th:href="@{/order/details(orderId=${o.id})}">view</a>

<!-- Will produce '/gtvg/order/3/details' (plus rewriting) -->

<a href="details.html" th:href="@{/order/{orderId}/details(orderId=${o.id})}">view</a>

**几点说明：**

**上例中URL最后的(orderId=${o.id})表示将括号内的内容作为URL参数处理，该语法避免使用字符串拼接，大大提高了可读性@{...}表达式中可以通过{orderId}访问Context中的orderId变量@{/order}是Context相关的相对路径，在渲染时会自动添加上当前Web应用的Context名字，假设context名字为app，那么结果应该是/app/order**

**3、字符串替换**

很多时候可能我们只需要对一大段文字中的某一处地方进行替换，可以通过字符串拼接操作完成：

<span th:text="'Welcome to our application, ' + ${user.name} + '!'">

**一种更简洁的方式是：**

<span th:text="|Welcome to our application, ${user.name}!|">

当然这种形式限制比较多，|…|中只能包含变量表达式${…}，不能包含其他常量、条件表达式等。

**4、运算符**

在表达式中可以使用各类算术运算符，例如+, -, \*, /, %

th:with="isEven=(${prodStat.count} % 2 == 0)"

逻辑运算符>, <, <=,>=，==,!=都可以使用，唯一需要注意的是使用<,>时需要用它的HTML转义符：

th:if="${prodStat.count} &gt; 1"

th:text="'Execution mode is ' + ( (${execMode} == 'dev')? 'Development' : 'Production')"

**二、常用的表达式**

**1、for循环**

**使用 th:each 标签**

<div class="row" >

<div th:each="url,lstat:${links}">

<div class="col-md-2" th:title="${url.description}" title="一个人，信你所信，为你所现" >

<strong th:text="${url.link\_name}">这个冬天不太冷</strong>

<a href="http://www.lrshuai.top" th:href="${url.link}" th:text="${url.link}" >http://www.lrshuai.top</a>

</div>

</div>

</div>

**lstat称作状态变量，属性有：**

* index:当前迭代对象的index（从0开始计算）
* count: 当前迭代对象的index(从1开始计算)
* size:被迭代对象的大小
* current:当前迭代变量
* even/odd:布尔值，当前循环是否是偶数/奇数（从0开始计算）
* first:布尔值，当前循环是否是第一个
* last:布尔值，当前循环是否是最后一个

**2、条件求值**

**If/Unless**

**demo**

<div class="row" >

<div th:each="url,lstat:${links}">

<div class="col-md-2" th:title="${url.description}" th:if="${lstat.index}%4 == 0" >

<strong th:text="${url.link\_name}">这个冬天不太冷</strong>

<a href="http://www.lrshuai.top" th:href="${url.link}" th:text="${url.link}">http://www.lrshuai.top</a>

</div>

<div class="col-md-2 col-md-offset-1" th:title="${url.description}" th:unless="${lstat.index}%4==0">

<strong th:text="${url.link\_name}">这个冬天不太冷</strong>

<a href="http://www.lrshuai.top" th:href="${url.link}" th:text="${url.link}" >http://www.lrshuai.top</a>

</div>

</div>

</div>

Thymeleaf中使用th:if和th:unless属性进行条件判断，上面的例子中，<div>标签只有在th:if中条件成立时才显示：

th:unless于th:if恰好相反，只有表达式中的条件不成立，才会显示其内容。

**Switch**

**Thymeleaf同样支持多路选择Switch结构：**

<div th:switch="${user.role}">

<p th:case="'admin'">User is an administrator</p>

<p th:case="#{roles.manager}">User is a manager</p>

</div>

**默认属性default可以用\*表示：**

<div th:switch="${user.role}">

<p th:case="'admin'">User is an administrator</p>

<p th:case="#{roles.manager}">User is a manager</p>

<p th:case="\*">User is some other thing</p>

</div>

**3、内嵌变量**

**为了模板更加易用，Thymeleaf还提供了一系列Utility对象（内置于Context中），可以通过#直接访问：**

* dates ： java.util.Date的功能方法类。
* calendars : 类似#dates，面向java.util.Calendar
* numbers : 格式化数字的功能方法类
* strings : 字符串对象的功能类
* objects: 对objects的功能类操作。
* bools: 对布尔值求值的功能方法。
* arrays：对数组的功能类方法。
* lists: 对lists功能类方法
* sets
* maps

**说说我常用得方法吧，太多了，你也不一定看完**

**(1)、字符串太多，显示…**

# 这里的含义是 如果 atc.text 这个变量多余200个字符，后面显示...

<p th:text="${#strings.abbreviate(atc.text,200)}">内容内容内容</p>

**(2)、数组判断是否为空**

<div th:if="${#lists.isEmpty(arrays)} " class="blog-article">

**(3)、request 获取绝对路径**

<img th:src="${#httpServletRequest.getContextPath()}+${atc.img}" src="/images/logo.jpg">

**常用th标签**

| **标签** | **说明** | **例子** |
| --- | --- | --- |
| th:id | 替换id | <input th:id="'xxx' + ${collect.id}"/> |
| th:text | 文本替换 | <p th:text="${collect.description}">description</p> |
| th:utext | 支持html的文本替换 | <p th:utext="${htmlcontent}">conten</p> |
| th:object | 替换对象 | <div th:object="${session.user}"> |
| th:value | 属性赋值 | <input th:value="${user.name}" /> |
| th:with | 变量赋值运算 | <div th:with="isEven=${prodStat.count}%2==0"></div> |
| th:style | 设置样式 | th:style="'display:' + @{(${sitrue} ? 'none' : 'inline-block')} + ''" |
| th:onclick | 点击事件 | th:onclick="'getCollect()'" |
| th:each | 属性赋值 | tr th:each="user,userStat:${users}"> |
| th:if | 判断条件 | <a th:if="${userId == collect.userId}" > |
| th:unless | 和th:if判断相反 | <a th:href="@{/login}" th:unless=${session.user != null}>Login</a> |
| th:href | 链接地址 | <a th:href="@{/login}" th:unless=${session.user != null}>Login</a> /> |
| th:switch | 多路选择 配合th:case 使用 | <div th:switch="${user.role}"> |
| th:case | th:switch的一个分支 | <p th:case="'admin'">User is an administrator</p> |
| th:fragment | 布局标签，定义一个代码片段，方便其它地方引用 | <div th:fragment="alert"> |
| th:include | 布局标签，替换内容到引入的文件 | <head th:include="layout :: htmlhead" th:with="title='xx'"></head> /> |
| th:replace | 布局标签，替换整个标签到引入的文件 | <div th:replace="fragments/header :: title"></div> |
| th:selected | selected选择框 选中 | th:selected="(${xxx.id} == ${configObj.dd})" |
| th:src | 图片类地址引入 | <img class="img-responsive" alt="App Logo" th:src="@{/img/logo.png}" /> |
| th:inline | 定义js脚本可以使用变量 | <script type="text/javascript" th:inline="javascript"> |
| th:action | 表单提交的地址 | <form action="subscribe.html" th:action="@{/subscribe}"> |
| th:remove | 删除某个属性 | <tr th:remove="all"> 1.all:删除包含标签和所有的孩子。2.body:不包含标记删除,但删除其所有的孩子。3.tag:包含标记的删除,但不删除它的孩子。4.all-but-first:删除所有包含标签的孩子,除了第一个。5.none:什么也不做。这个值是有用的动态评估。 |
| th:attr | 设置标签属性，多个属性可以用逗号分隔 | 比如<p th:attr="src=@{/image/aa.jpg},title=${title}">内容</p>，这样如果${title}=’这个是title’ 则结果就是<p src="/image/aa.jpg" title="这个是title">内容</p> |

html 有的，它几乎都有相对应的标签

**下面是一组的API**

**日期: #dates**

/\*

\* ======================================================================

\* See javadoc API for class org.thymeleaf.expression.Dates

\* ======================================================================

\*/

/\*

\* Format date with the standard locale format

\* Also works with arrays, lists or sets

\*/

${#dates.format(date)}

${#dates.arrayFormat(datesArray)}

${#dates.listFormat(datesList)}

${#dates.setFormat(datesSet)}

/\*

\* Format date with the ISO8601 format

\* Also works with arrays, lists or sets

\*/

${#dates.formatISO(date)}

${#dates.arrayFormatISO(datesArray)}

${#dates.listFormatISO(datesList)}

${#dates.setFormatISO(datesSet)}

/\*

\* Format date with the specified pattern

\* Also works with arrays, lists or sets

\*/

${#dates.format(date, 'dd/MMM/yyyy HH:mm')}

${#dates.arrayFormat(datesArray, 'dd/MMM/yyyy HH:mm')}

${#dates.listFormat(datesList, 'dd/MMM/yyyy HH:mm')}

${#dates.setFormat(datesSet, 'dd/MMM/yyyy HH:mm')}

/\*

\* Obtain date properties

\* Also works with arrays, lists or sets

\*/

${#dates.day(date)} // also arrayDay(...), listDay(...), etc.

${#dates.month(date)} // also arrayMonth(...), listMonth(...), etc.

${#dates.monthName(date)} // also arrayMonthName(...), listMonthName(...), etc.

${#dates.monthNameShort(date)} // also arrayMonthNameShort(...), listMonthNameShort(...), etc.

${#dates.year(date)} // also arrayYear(...), listYear(...), etc.

${#dates.dayOfWeek(date)} // also arrayDayOfWeek(...), listDayOfWeek(...), etc.

${#dates.dayOfWeekName(date)} // also arrayDayOfWeekName(...), listDayOfWeekName(...), etc.

${#dates.dayOfWeekNameShort(date)} // also arrayDayOfWeekNameShort(...), listDayOfWeekNameShort(...), etc.

${#dates.hour(date)} // also arrayHour(...), listHour(...), etc.

${#dates.minute(date)} // also arrayMinute(...), listMinute(...), etc.

${#dates.second(date)} // also arraySecond(...), listSecond(...), etc.

${#dates.millisecond(date)} // also arrayMillisecond(...), listMillisecond(...), etc.

/\*

\* Create date (java.util.Date) objects from its components

\*/

${#dates.create(year,month,day)}

${#dates.create(year,month,day,hour,minute)}

${#dates.create(year,month,day,hour,minute,second)}

${#dates.create(year,month,day,hour,minute,second,millisecond)}

/\*

\* Create a date (java.util.Date) object for the current date and time

\*/

${#dates.createNow()}

${#dates.createNowForTimeZone()}

/\*

\* Create a date (java.util.Date) object for the current date (time set to 00:00)

\*/

${#dates.createToday()}

${#dates.createTodayForTimeZone()}

**数字:#numbers**

/\*

\* ======================================================================

\* See javadoc API for class org.thymeleaf.expression.Numbers

\* ======================================================================

\*/

/\*

\* ==========================

\* Formatting integer numbers

\* ==========================

\*/

/\*

\* Set minimum integer digits.

\* Also works with arrays, lists or sets

\*/

${#numbers.formatInteger(num,3)}

${#numbers.arrayFormatInteger(numArray,3)}

${#numbers.listFormatInteger(numList,3)}

${#numbers.setFormatInteger(numSet,3)}

/\*

\* Set minimum integer digits and thousands separator:

\* 'POINT', 'COMMA', 'WHITESPACE', 'NONE' or 'DEFAULT' (by locale).

\* Also works with arrays, lists or sets

\*/

${#numbers.formatInteger(num,3,'POINT')}

${#numbers.arrayFormatInteger(numArray,3,'POINT')}

${#numbers.listFormatInteger(numList,3,'POINT')}

${#numbers.setFormatInteger(numSet,3,'POINT')}

/\*

\* ==========================

\* Formatting decimal numbers

\* ==========================

\*/

/\*

\* Set minimum integer digits and (exact) decimal digits.

\* Also works with arrays, lists or sets

\*/

${#numbers.formatDecimal(num,3,2)}

${#numbers.arrayFormatDecimal(numArray,3,2)}

${#numbers.listFormatDecimal(numList,3,2)}

${#numbers.setFormatDecimal(numSet,3,2)}

/\*

\* Set minimum integer digits and (exact) decimal digits, and also decimal separator.

\* Also works with arrays, lists or sets

\*/

${#numbers.formatDecimal(num,3,2,'COMMA')}

${#numbers.arrayFormatDecimal(numArray,3,2,'COMMA')}

${#numbers.listFormatDecimal(numList,3,2,'COMMA')}

${#numbers.setFormatDecimal(numSet,3,2,'COMMA')}

/\*

\* Set minimum integer digits and (exact) decimal digits, and also thousands and

\* decimal separator.

\* Also works with arrays, lists or sets

\*/

${#numbers.formatDecimal(num,3,'POINT',2,'COMMA')}

${#numbers.arrayFormatDecimal(numArray,3,'POINT',2,'COMMA')}

${#numbers.listFormatDecimal(numList,3,'POINT',2,'COMMA')}

${#numbers.setFormatDecimal(numSet,3,'POINT',2,'COMMA')}

/\*

\* =====================

\* Formatting currencies

\* =====================

\*/

${#numbers.formatCurrency(num)}

${#numbers.arrayFormatCurrency(numArray)}

${#numbers.listFormatCurrency(numList)}

${#numbers.setFormatCurrency(numSet)}

/\*

\* ======================

\* Formatting percentages

\* ======================

\*/

${#numbers.formatPercent(num)}

${#numbers.arrayFormatPercent(numArray)}

${#numbers.listFormatPercent(numList)}

${#numbers.setFormatPercent(numSet)}

/\*

\* Set minimum integer digits and (exact) decimal digits.

\*/

${#numbers.formatPercent(num, 3, 2)}

${#numbers.arrayFormatPercent(numArray, 3, 2)}

${#numbers.listFormatPercent(numList, 3, 2)}

${#numbers.setFormatPercent(numSet, 3, 2)}

/\*

\* ===============

\* Utility methods

\* ===============

\*/

/\*

\* Create a sequence (array) of integer numbers going

\* from x to y

\*/

${#numbers.sequence(from,to)}

${#numbers.sequence(from,to,step)}

**字符串:#strings**

/\*

\* Null-safe toString()

\*/

${#strings.toString(obj)} // also array\*, list\* and set\*

/\*

\* Check whether a String is empty (or null). Performs a trim() operation before check

\* Also works with arrays, lists or sets

\*/

${#strings.isEmpty(name)}

${#strings.arrayIsEmpty(nameArr)}

${#strings.listIsEmpty(nameList)}

${#strings.setIsEmpty(nameSet)}

/\*

\* Perform an 'isEmpty()' check on a string and return it if false, defaulting to

\* another specified string if true.

\* Also works with arrays, lists or sets

\*/

${#strings.defaultString(text,default)}

${#strings.arrayDefaultString(textArr,default)}

${#strings.listDefaultString(textList,default)}

${#strings.setDefaultString(textSet,default)}

/\*

\* Check whether a fragment is contained in a String

\* Also works with arrays, lists or sets

\*/

${#strings.contains(name,'ez')} // also array\*, list\* and set\*

${#strings.containsIgnoreCase(name,'ez')} // also array\*, list\* and set\*

/\*

\* Check whether a String starts or ends with a fragment

\* Also works with arrays, lists or sets

\*/

${#strings.startsWith(name,'Don')} // also array\*, list\* and set\*

${#strings.endsWith(name,endingFragment)} // also array\*, list\* and set\*

/\*

\* Substring-related operations

\* Also works with arrays, lists or sets

\*/

${#strings.indexOf(name,frag)} // also array\*, list\* and set\*

${#strings.substring(name,3,5)} // also array\*, list\* and set\*

${#strings.substringAfter(name,prefix)} // also array\*, list\* and set\*

${#strings.substringBefore(name,suffix)} // also array\*, list\* and set\*

${#strings.replace(name,'las','ler')} // also array\*, list\* and set\*

/\*

\* Append and prepend

\* Also works with arrays, lists or sets

\*/

${#strings.prepend(str,prefix)} // also array\*, list\* and set\*

${#strings.append(str,suffix)} // also array\*, list\* and set\*

/\*

\* Change case

\* Also works with arrays, lists or sets

\*/

${#strings.toUpperCase(name)} // also array\*, list\* and set\*

${#strings.toLowerCase(name)} // also array\*, list\* and set\*

/\*

\* Split and join

\*/

${#strings.arrayJoin(namesArray,',')}

${#strings.listJoin(namesList,',')}

${#strings.setJoin(namesSet,',')}

${#strings.arraySplit(namesStr,',')} // returns String[]

${#strings.listSplit(namesStr,',')} // returns List<String>

${#strings.setSplit(namesStr,',')} // returns Set<String>

/\*

\* Trim

\* Also works with arrays, lists or sets

\*/

${#strings.trim(str)} // also array\*, list\* and set\*

/\*

\* Compute length

\* Also works with arrays, lists or sets

\*/

${#strings.length(str)} // also array\*, list\* and set\*

/\*

\* Abbreviate text making it have a maximum size of n. If text is bigger, it

\* will be clipped and finished in "..."

\* Also works with arrays, lists or sets

\*/

${#strings.abbreviate(str,10)} // also array\*, list\* and set\*

/\*

\* Convert the first character to upper-case (and vice-versa)

\*/

${#strings.capitalize(str)} // also array\*, list\* and set\*

${#strings.unCapitalize(str)} // also array\*, list\* and set\*

/\*

\* Convert the first character of every word to upper-case

\*/

${#strings.capitalizeWords(str)} // also array\*, list\* and set\*

${#strings.capitalizeWords(str,delimiters)} // also array\*, list\* and set\*

/\*

\* Escape the string

\*/

${#strings.escapeXml(str)} // also array\*, list\* and set\*

${#strings.escapeJava(str)} // also array\*, list\* and set\*

${#strings.escapeJavaScript(str)} // also array\*, list\* and set\*

${#strings.unescapeJava(str)} // also array\*, list\* and set\*

${#strings.unescapeJavaScript(str)} // also array\*, list\* and set\*

/\*

\* Null-safe comparison and concatenation

\*/

${#strings.equals(first, second)}

${#strings.equalsIgnoreCase(first, second)}

${#strings.concat(values...)}

${#strings.concatReplaceNulls(nullValue, values...)}

/\*

\* Random

\*/

${#strings.randomAlphanumeric(count)}

**布尔:#bools**

/\*

\* Evaluate a condition in the same way that it would be evaluated in a th:if tag

\* (see conditional evaluation chapter afterwards).

\* Also works with arrays, lists or sets

\*/

${#bools.isTrue(obj)}

${#bools.arrayIsTrue(objArray)}

${#bools.listIsTrue(objList)}

${#bools.setIsTrue(objSet)}

/\*

\* Evaluate with negation

\* Also works with arrays, lists or sets

\*/

${#bools.isFalse(cond)}

${#bools.arrayIsFalse(condArray)}

${#bools.listIsFalse(condList)}

${#bools.setIsFalse(condSet)}

/\*

\* Evaluate and apply AND operator

\* Receive an array, a list or a set as parameter

\*/

${#bools.arrayAnd(condArray)}

${#bools.listAnd(condList)}

${#bools.setAnd(condSet)}

/\*

\* Evaluate and apply OR operator

\* Receive an array, a list or a set as parameter

\*/

${#bools.arrayOr(condArray)}

${#bools.listOr(condList)}

${#bools.setOr(condSet)}

**数组 :#arrays**

/\*

\* Converts to array, trying to infer array component class.

\* Note that if resulting array is empty, or if the elements

\* of the target object are not all of the same class,

\* this method will return Object[].

\*/

${#arrays.toArray(object)}

/\*

\* Convert to arrays of the specified component class.

\*/

${#arrays.toStringArray(object)}

${#arrays.toIntegerArray(object)}

${#arrays.toLongArray(object)}

${#arrays.toDoubleArray(object)}

${#arrays.toFloatArray(object)}

${#arrays.toBooleanArray(object)}

/\*

\* Compute length

\*/

${#arrays.length(array)}

/\*

\* Check whether array is empty

\*/

${#arrays.isEmpty(array)}

/\*

\* Check if element or elements are contained in array

\*/

${#arrays.contains(array, element)}

${#arrays.containsAll(array, elements)}