北京郵電大學
Beijing University of Posts and Telecommunications

Web Programming

Web开发技术基础

第6章 ThymeLeaf

计算机学院

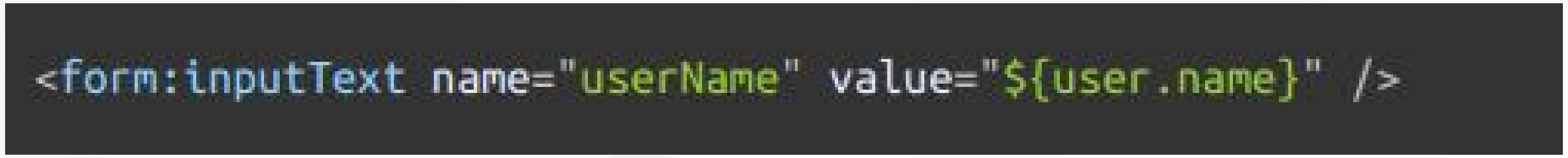
授课人：王尊亮

第6章 Thymeleaf

6.1 Thymeleaf简介

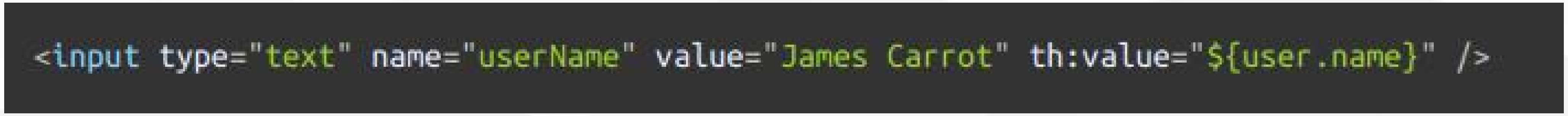
Thymeleaf：Java服务端的模板引擎，不新增标签，采用拓展属性（th:xx）去跟服务端进行数据交互，保留原始页面风格，使用浏览器直接打开，相当于打开原生页面，简洁漂亮、容易理解，完美支持HTML5，给前端人员也带来一定的便利。

JSP：



```
<form:inputText name="userName" value="${user.name}" />
```

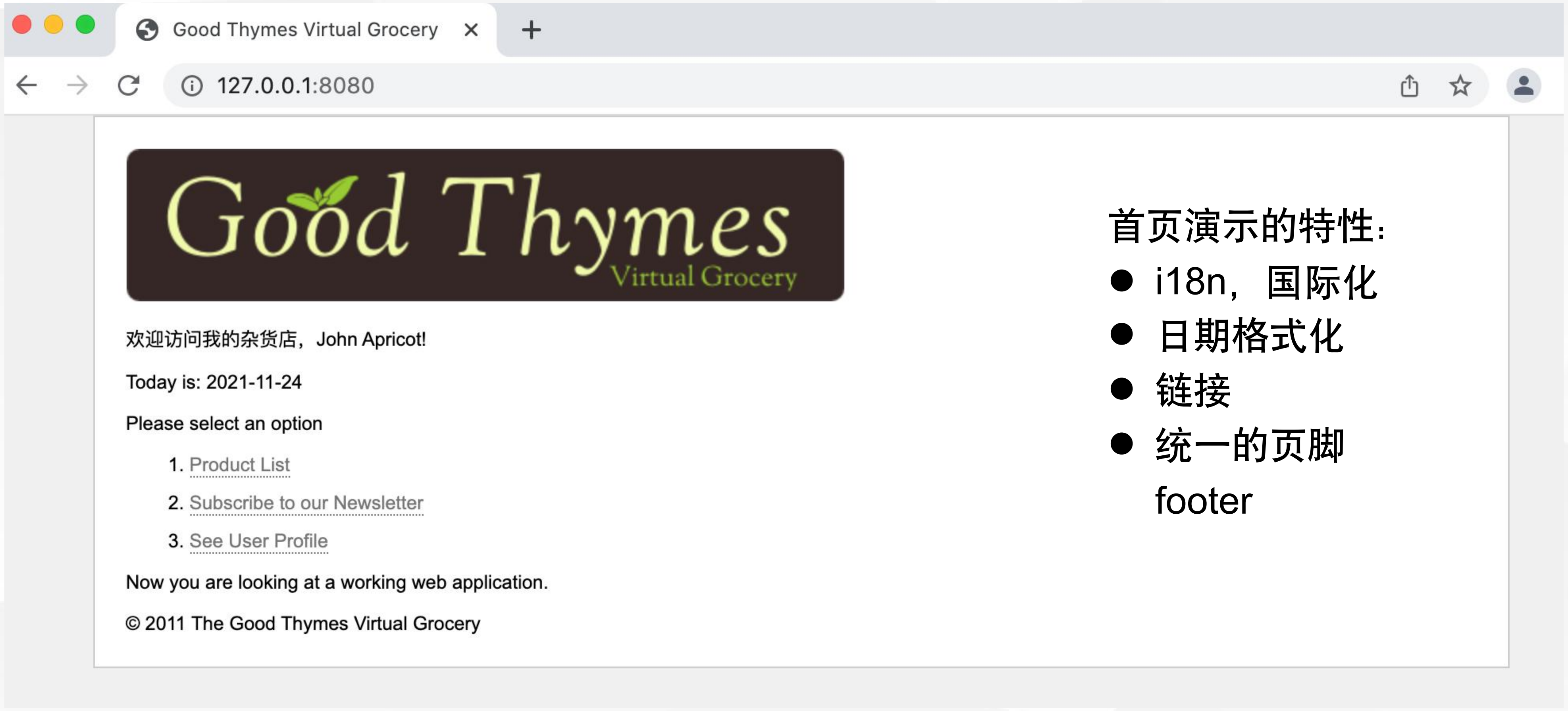
Thymeleaf：



```
<input type="text" name="userName" value="James Carrot" th:value="${user.name}" />
```

6.1 Thymeleaf简介

Thymeleaf 示例项目“The Good Thymes Virtual Grocery”



欢迎访问我的杂货店，John Apricot!

Today is: 2021-11-24

Please select an option

1. Product List
2. Subscribe to our Newsletter
3. See User Profile

Now you are looking at a working web application.

© 2011 The Good Thymes Virtual Grocery

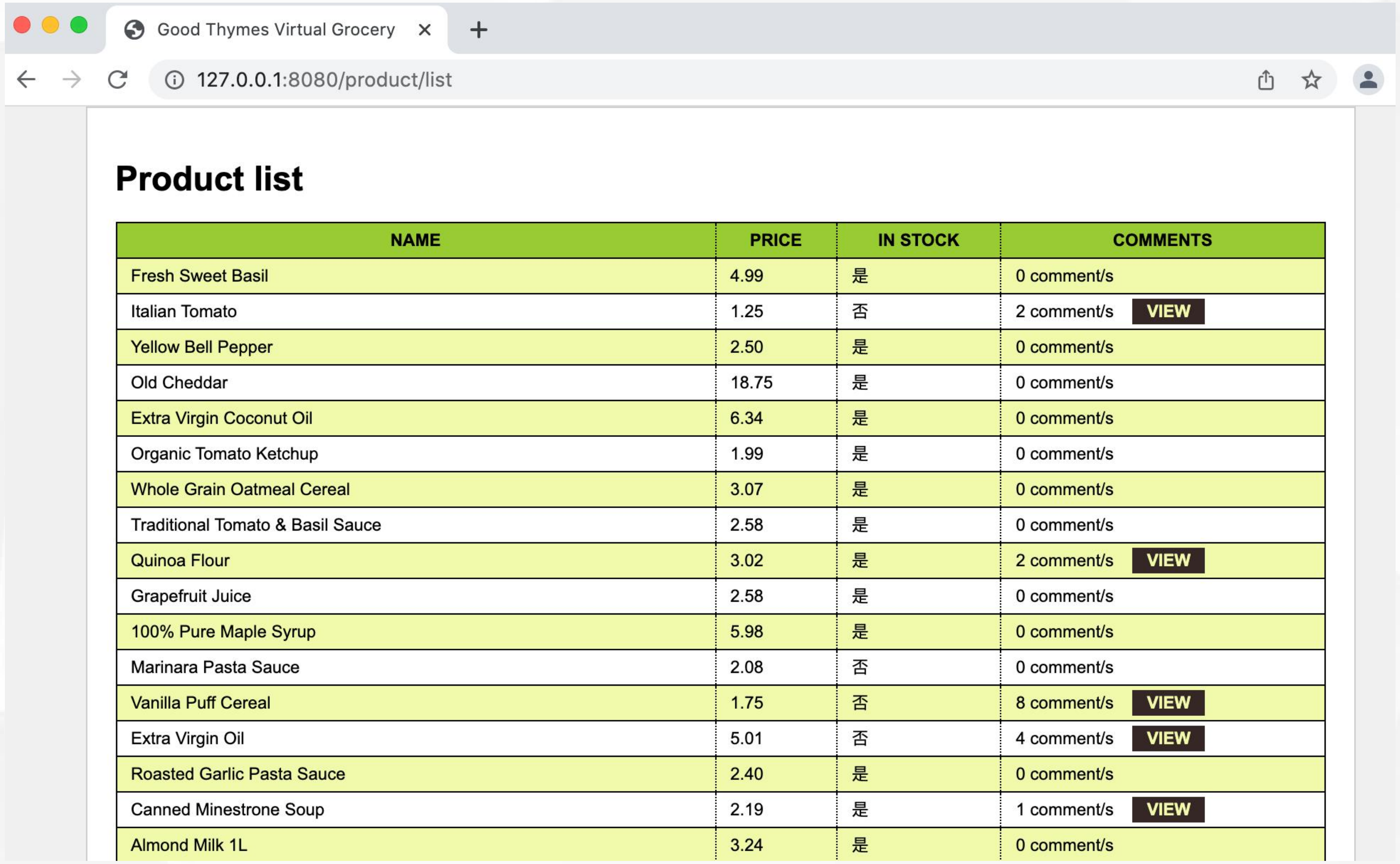
首页演示的特性：

- i18n，国际化
- 日期格式化
- 链接
- 统一的页脚 footer

https://gitee.com/buptnetwork/thymeleafexamples

6.1 Thymeleaf简介

Thymeleaf 示例项目“The Good Thymes Virtual Grocery”



Product list

NAME	PRICE	IN STOCK	COMMENTS
Fresh Sweet Basil	4.99	是	0 comment/s
Italian Tomato	1.25	否	2 comment/s VIEW
Yellow Bell Pepper	2.50	是	0 comment/s
Old Cheddar	18.75	是	0 comment/s
Extra Virgin Coconut Oil	6.34	是	0 comment/s
Organic Tomato Ketchup	1.99	是	0 comment/s
Whole Grain Oatmeal Cereal	3.07	是	0 comment/s
Traditional Tomato & Basil Sauce	2.58	是	0 comment/s
Quinoa Flour	3.02	是	2 comment/s VIEW
Grapefruit Juice	2.58	是	0 comment/s
100% Pure Maple Syrup	5.98	是	0 comment/s
Marinara Pasta Sauce	2.08	否	0 comment/s
Vanilla Puff Cereal	1.75	否	8 comment/s VIEW
Extra Virgin Oil	5.01	否	4 comment/s VIEW
Roasted Garlic Pasta Sauce	2.40	是	0 comment/s
Canned Minestrone Soup	2.19	是	1 comment/s VIEW
Almond Milk 1L	3.24	是	0 comment/s

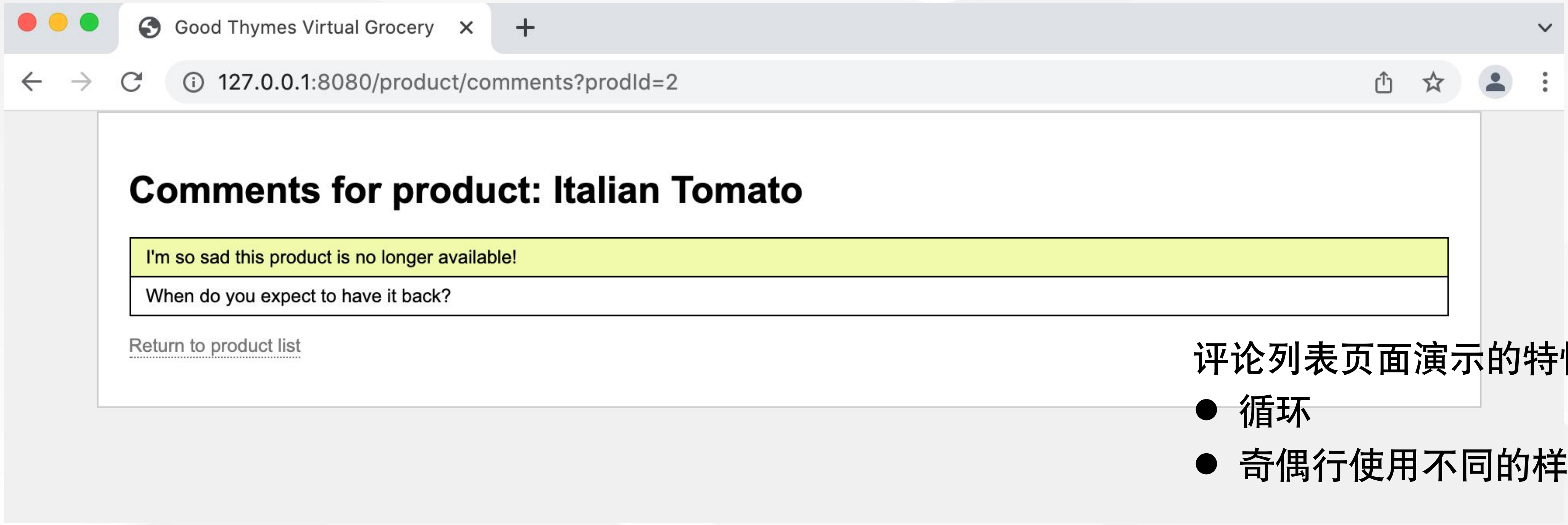
商品列表页面演示的特性：

- 循环
- 条件控制，评论数>0则显示view按钮
- 奇偶行使用不同的样式class

https://gitee.com/buptnetwork/thymeleafexamples

6.1 Thymeleaf简介

Thymeleaf 示例项目“The Good Thymes Virtual Grocery”


Good Thymes Virtual Grocery
127.0.0.1:8080/product/comments?prodId=2

Comments for product: Italian Tomato

I'm so sad this product is no longer available!
When do you expect to have it back?

Return to product list

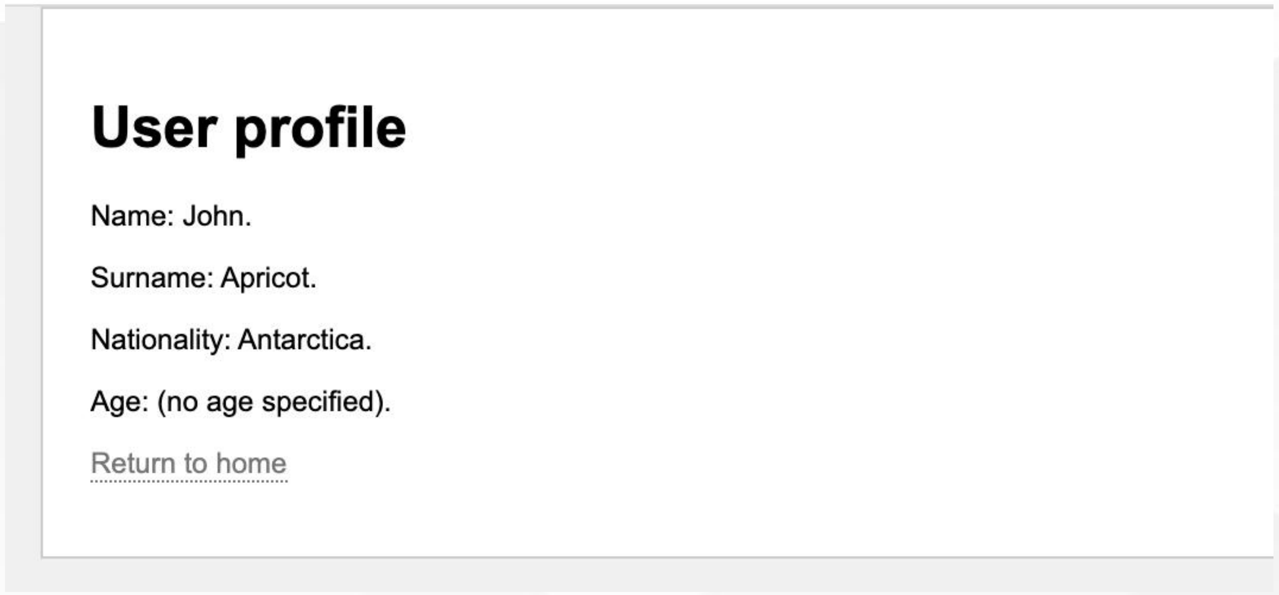
评论列表页面演示的特性：

- 循环
- 奇偶行使用不同的样式class

https://gitee.com/buptnetwork/thymeleafexamples

6.1 Thymeleaf简介

Thymeleaf 示例项目“The Good Thymes Virtual Grocery”



User profile

Name: John.

Surname: Apricot.

Nationality: Antarctica.

Age: (no age specified).

Return to home

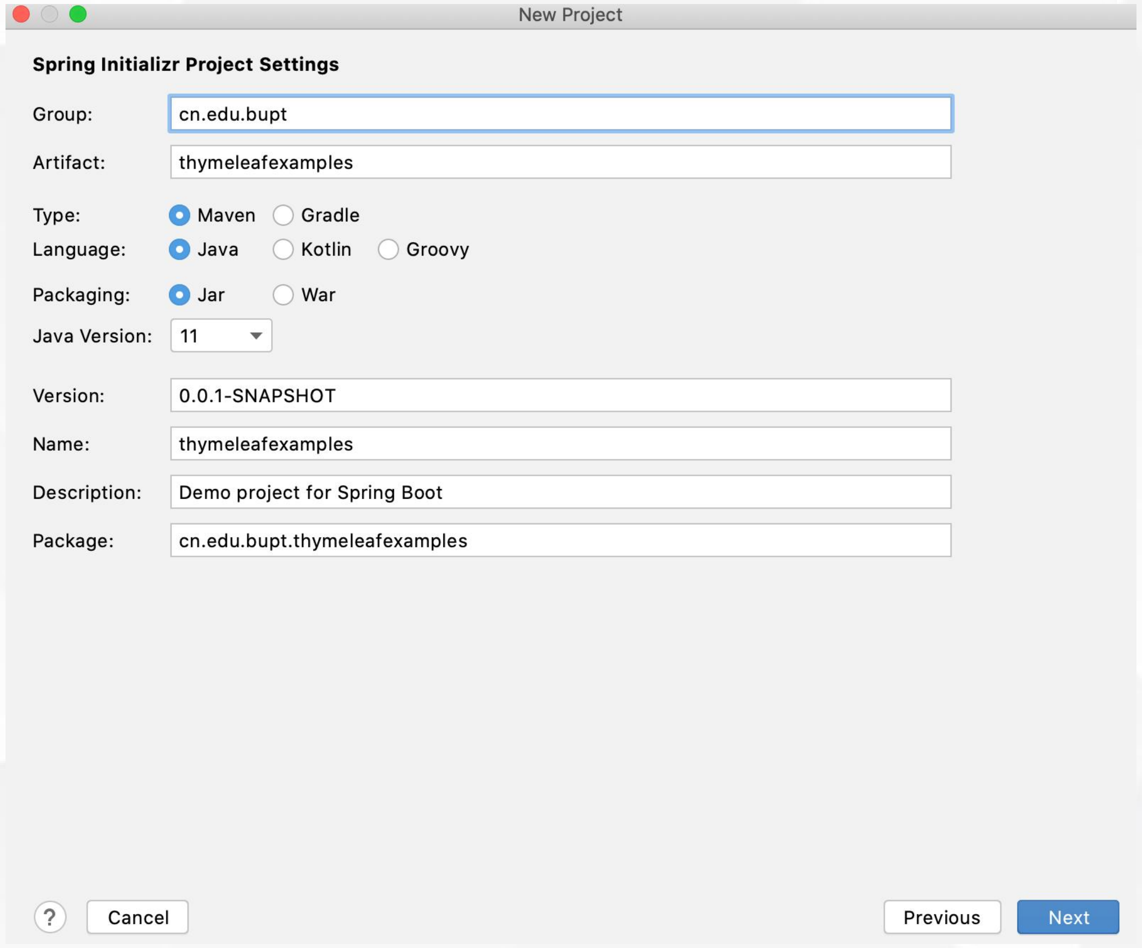
profile页面演示的特性：

- 如何更方便的展示一个对象的多个属性
- 条件表达式

https://gitee.com/buptnetwork/thymeleafexamples

6.1 Thymeleaf简介

Thymeleaf 示例项目“The Good Thymes Virtual Grocery”



New Project

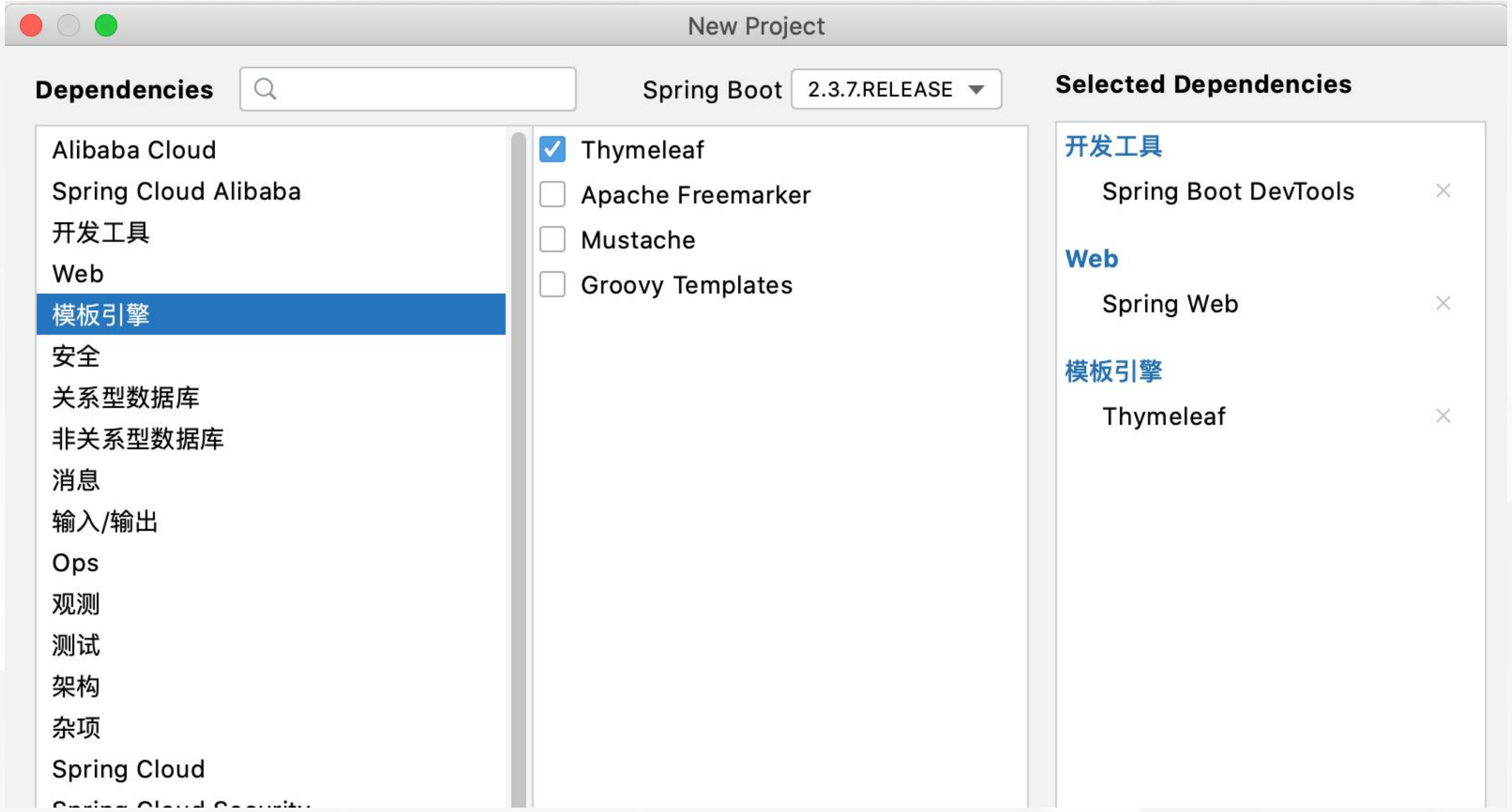
Spring Initializr Project Settings

Group:	cn.edu.bupt
Artifact:	thymeleafexamples
Type:	[✓ Maven] Gradle
Language:	[✓ Java] Kotlin Groovy
Packaging:	[✓ Jar] War
Java Version:	11
Version:	0.0.1-SNAPSHOT
Name:	thymeleafexamples
Description:	Demo project for Spring Boot
Package:	cn.edu.bupt.thymeleafexamples

https://gitee.com/buptnetwork/thymeleafexamples

6.1 Thymeleaf简介

Thymeleaf 示例项目“The Good Thymes Virtual Grocery”


New Project

Dependencies

Spring Boot 2.3.7.RELEASE

Selected Dependencies

- Alibaba Cloud
- Spring Cloud Alibaba
- 开发工具
- Web
- 模板引擎
- 安全
- 关系型数据库
- 非关系型数据库
- 消息
- 输入/输出
- Ops
- 观测
- 测试
- 架构
- 杂项
- Spring Cloud

- [x] Thymeleaf
- [ ] Apache Freemarker
- [ ] Mustache
- [ ] Groovy Templates

开发工具
Spring Boot DevTools

Web
Spring Web

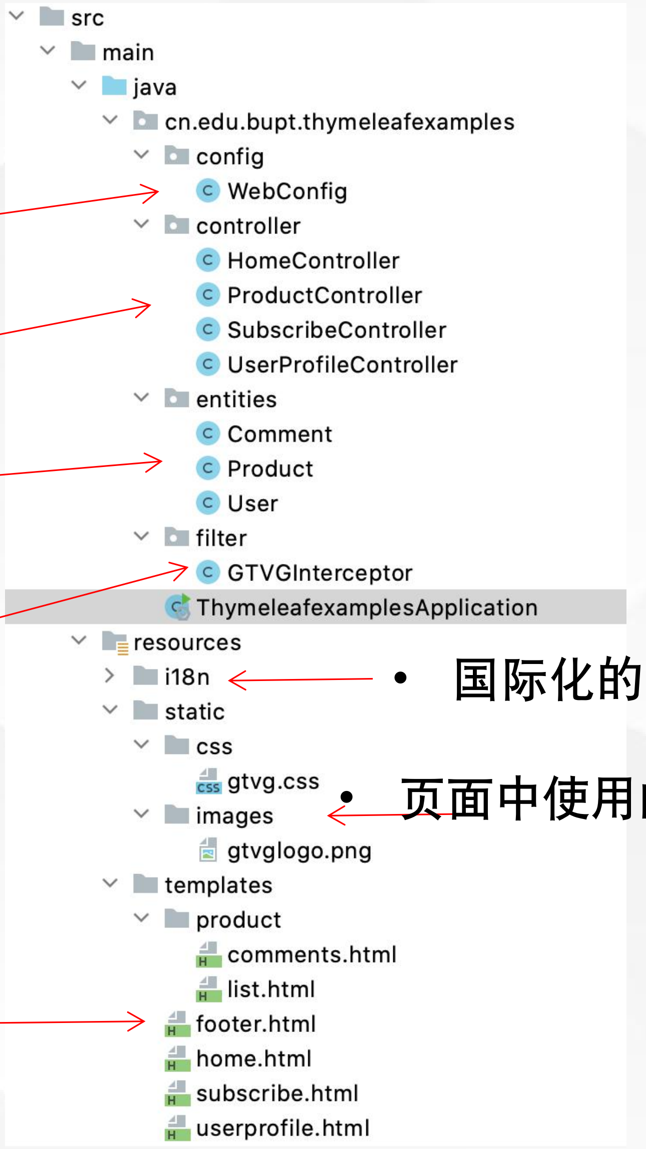
模板引擎
Thymeleaf

https://gitee.com/buptnetwork/thymeleafexamples

6.1 Thymeleaf简介

Thymeleaf 示例项目“The Good Thymes Virtual Grocery”

- 额外添加了一个Spring MVC的配置类，为项目添加了一个拦截器，和thymeleaf模板没有必然联系。
- 处理各功能模块请求的控制器类
- 本项目使用的各个POJO（简单JAVA对象）类
- 拦截器，在本项目中用于向session中添加一个user属性，值为一个User对象。
- 国际化的资源属性文件
- 页面中使用的静态资源文件
- 页面模板文件



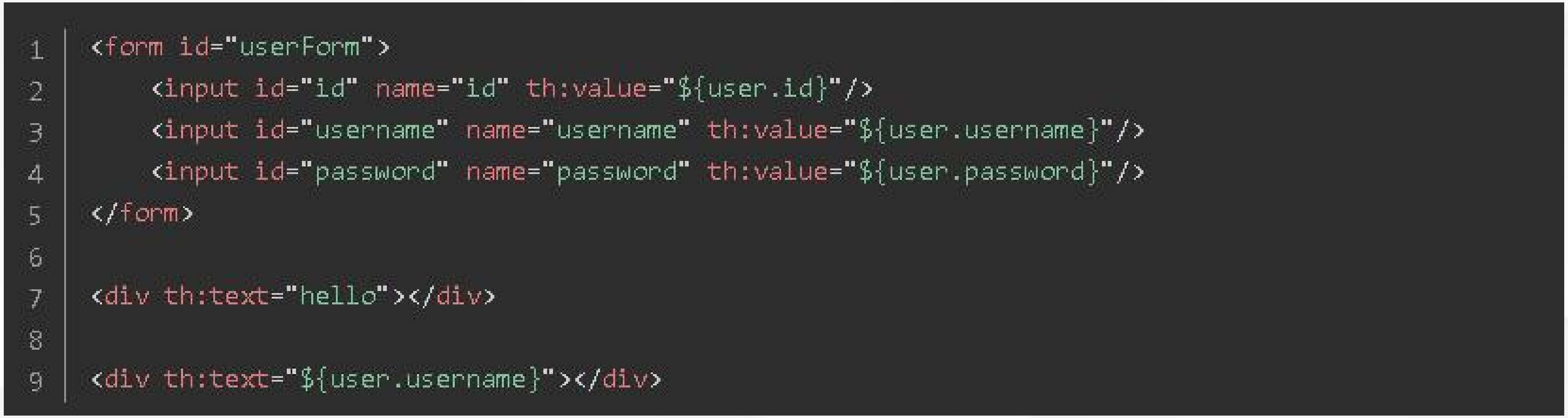
```
src
  main
    java
      cn.edu.bupt.thymeleafexamples
        config
          WebConfig
        controller
          HomeController
          ProductController
          SubscribeController
          UserProfileController
        entities
          Comment
          Product
          User
        filter
          GTVGInterceptor
        ThymeleafexamplesApplication
    resources
      i18n
      static
        css
          gtvg.css
        images
          gtvglogo.png
      templates
        product
          comments.html
          list.html
        footer.html
        home.html
        subscribe.html
        userprofile.html
```

https://gitee.com/buptnetwork/thymeleafexamples

6.2 基础语法

文档地址：https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#standard-expression-syntax

变量表达式 **${}** ： 直接使用th:xx = "${}" 将html元素的xx属性设置为变量的值 。例如：



```
<form id="userForm">
    <input id="id" name="id" th:value="${user.id}"/>
    <input id="username" name="username" th:value="${user.username}"/>
    <input id="password" name="password" th:value="${user.password}"/>
</form>

<div th:text="hello"></div>

<div th:text="${user.username}"></div>
```

6.2 基础语法

文档地址：https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#standard-expression-syntax

**变量表达式 ${}** ： 直接使用${} 获取对象属性 。例如：

```
/*
 * Access to properties using the point (.). Equivalent to calling property getters.
 * 使用(.)获取对象的属性
 */
${person.father.name}

/*
 * Access to properties can also be made by using brackets ([]) and writing
 * the name of the property as a variable or between single quotes.
 * 还可以使用['属性名']获取对象的属性
 */
${person['father']['name']}
```

6.2 基础语法

文档地址：https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#standard-expression-syntax

变量表达式 ${} ： 直接使用${} 获取对象属性 。例如：

```
/*
 * If the object is a map, both dot and bracket syntax will be equivalent to
 * executing a call on its get(...) method.
 * 如果对象是map，可以用.或[]调用其get方法
 */
${countriesByCode.ES}
${personsByName['Stephen Zucchini'].age}

/*
 * Indexed access to arrays or collections is also performed with brackets,
 * writing the index without quotes.
 * 使用[序号]访问一个数组或集合中的元素
 */
${personsArray[0].name}
```

6.2 基础语法

文档地址：https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#standard-expression-syntax

**变量表达式 ${}** ：直接使用${} 获取对象属性 。例如：

```
/*
 * Methods can be called, even with arguments.
 * 在变量表达式中还可以调用对象的方法，即使方法需要参数
 */
${person.createCompleteName()}
${person.createCompleteNameWithSeparator('-')}
```

6.2 基础语法

选择变量表达式 *{} ： 首先通过th:object 获取对象， 然后使用 "*{}"获取对象属性。

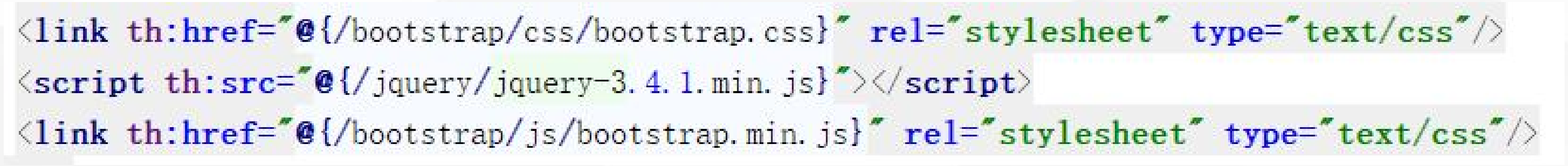
```
<div th:object="${session.user}">
  <p>Name: <span th:text="*{firstName}">Sebastian</span>.</p>
  <p>Surname: <span th:text="*{lastName}">Pepper</span>.</p>
  <p>Nationality: <span th:text="*{nationality}">Saturn</span>.</p>
</div>
```

等价于下面的表达式

```
<div>
  <p>Name: <span th:text="${session.user.firstName}">Sebastian</span>.</p>
  <p>Surname: <span th:text="${session.user.lastName}">Pepper</span>.</p>
  <p>Nationality: <span th:text="${session.user.nationality}">Saturn</span>.</p>
</div>
```

6.2 基础语法

链接表达式 @{} ：通过链接表达式@{}直接拿到应用路径，然后拼接静态资源路径。例如：



```
<link th:href="@{/bootstrap/css/bootstrap.css}" rel="stylesheet" type="text/css"/>
<script th:src="@{/jquery/jquery-3.4.1.min.js}"></script>
<link th:href="@{/bootstrap/js/bootstrap.min.js}" rel="stylesheet" type="text/css"/>
```

绝对URL：http://www.thymeleaf.org

相对URL：

- 相对页面地址：user/login.html
- 相对上下文地址：/itemdetails?id=3，将自动在前面添加上下文path
- 相对服务地址：~/billing/processInvoice，不再自动添加上下文path，可以用来访问本服务上的其它上下文页面
- 相对协议地址：//code.jquery.com/jquery-2.0.3.min.js

6.2 基础语法

链接表达式 @{} ：通过链接表达式@{}直接拿到应用路径，然后拼接静态资源路径。例如：

```
<!-- 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>
```

```
@{/order/process(execId=${execId},execType='FAST')}
```

url携带的参数

多个参数用“，”分割

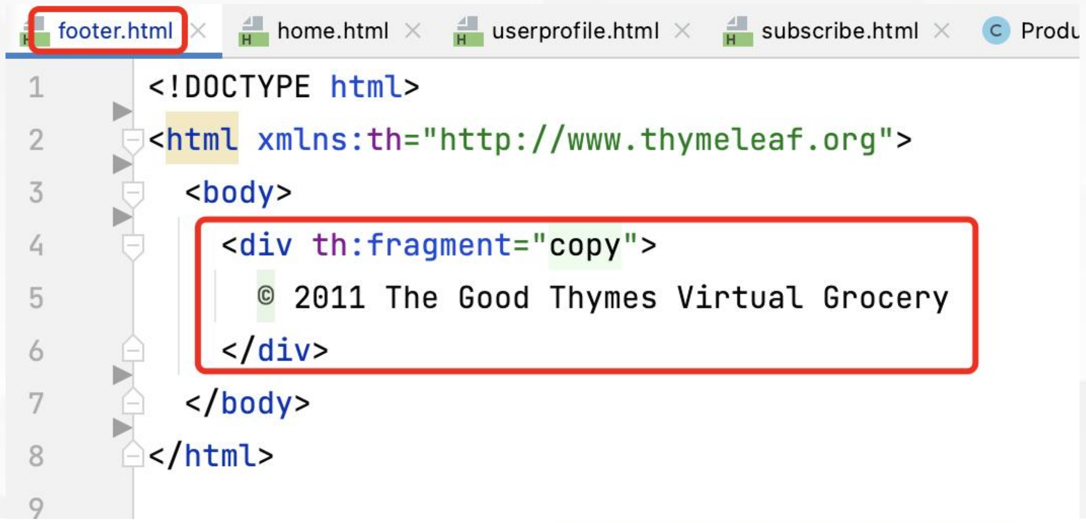
6.2 基础语法

片段表达式 ~{} ： 首先通过th:fragment定制片段， 然后通过th:replace 填写片段路径和片段名。

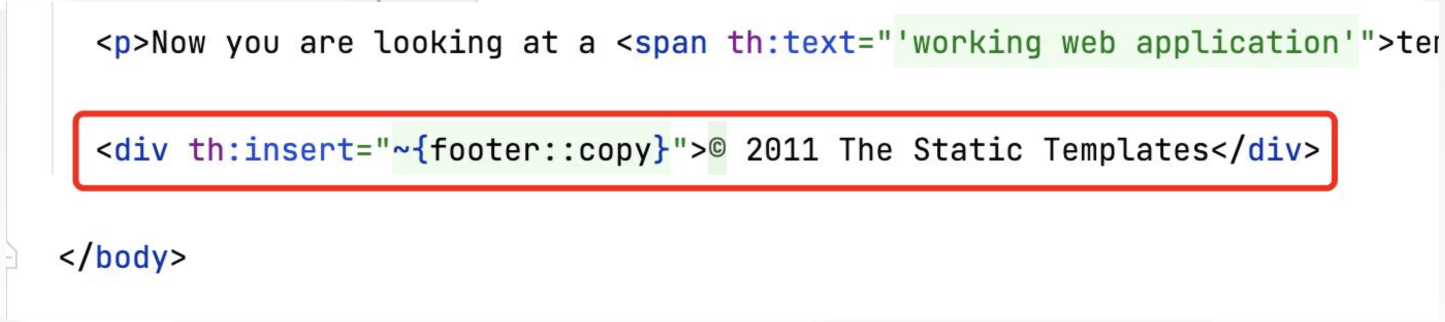
~{ viewName } 表示引入完整页面

~{ viewName ::selector} 表示在指定页面寻找片段 其中selector可为片段名、jquery选择器等

~{ ::selector} 表示在当前页寻找



```html
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
  <body>
    <div th:fragment="copy">
      © 2011 The Good Thymes Virtual Grocery
    </div>
  </body>
</html>
```



```html
  <p>Now you are looking at a <span th:text="'working web application'">ter

  <div th:insert="~{footer::copy}">© 2011 The Static Templates</div>

</body>
```

6.2 基础语法

消息表达式#{} ： #{msg} 用于获取国际化语言翻译值。

```
<title th:text="#{user.title}"></title>
```

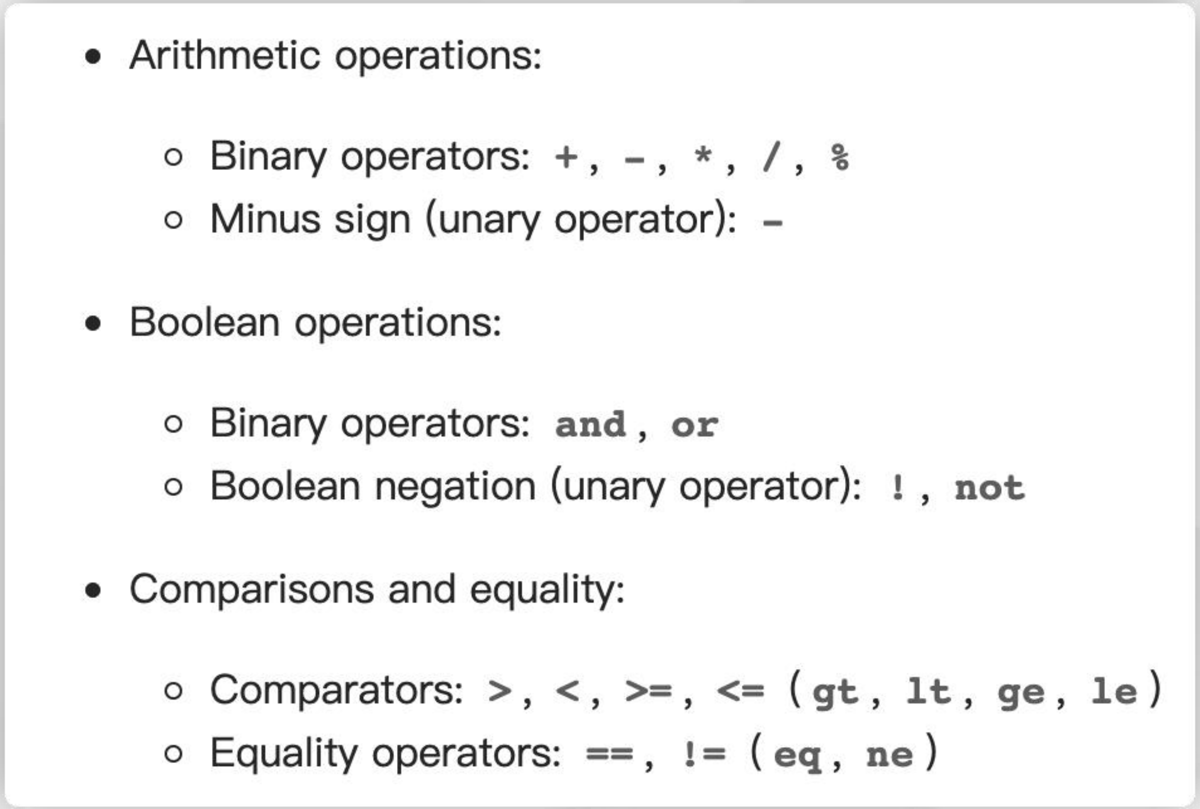
其它表达式

在基础语法中，默认支持字符串连接、数学运算、布尔逻辑和三目运算等。

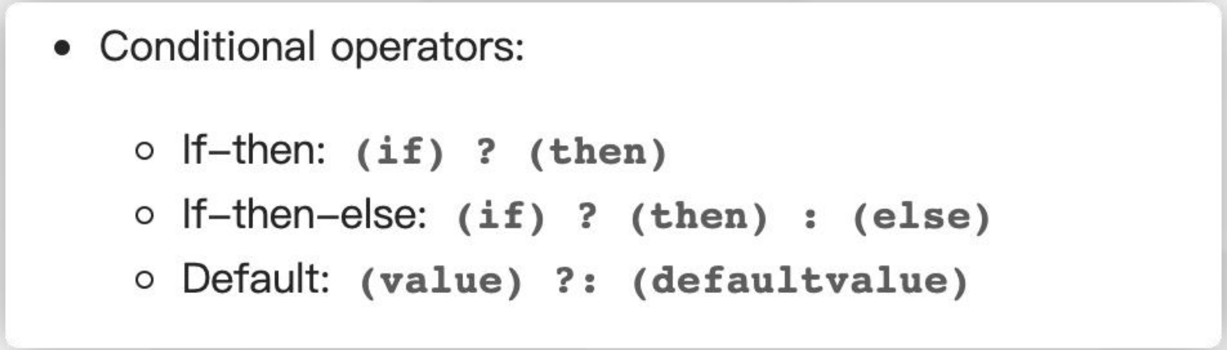
```
<input name="name" th:value="${'I am '+(user.name!=null?user.name:'NoBody')}"/>
```

```
<p>Age: <span th:text="*{age != null}? *{age} : '(no age specified)'">27</span>.</p>
```

6.2 基础语法



- Arithmetic operations:
  - Binary operators: `+`, `-`, `*`, `/`, `%`
  - Minus sign (unary operator): `-`
- Boolean operations:
  - Binary operators: `and`, `or`
  - Boolean negation (unary operator): `!`, `not`
- Comparisons and equality:
  - Comparators: `>`, `<`, `>=`, `<=` (`gt`, `lt`, `ge`, `le`)
  - Equality operators: `==`, `!=` (`eq`, `ne`)



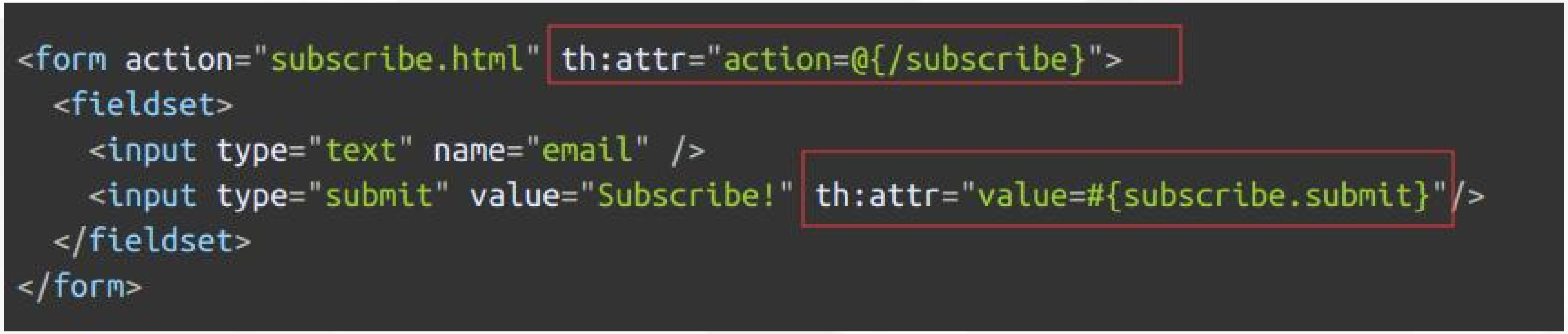
- Conditional operators:
  - If-then: `(if) ? (then)`
  - If-then-else: `(if) ? (then) : (else)`
  - Default: `(value) ?: (defaultvalue)`

'User is of type ' + (${user.isAdmin()} ? 'Administrator' : (${user.type} ?: 'Unknown'))

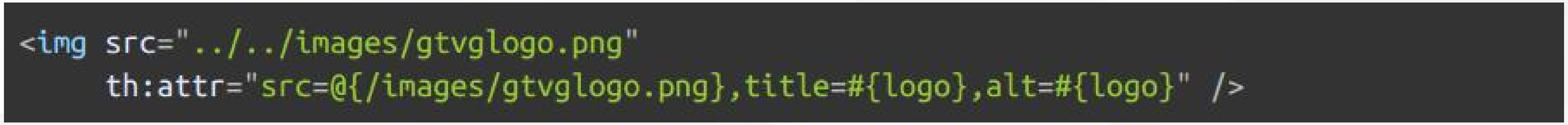
6.2 基础语法

设置任意属性的值th:attr

文档：https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#setting-attribute-values



```
<form action="subscribe.html" th:attr="action=@{/subscribe}">
  <fieldset>
    <input type="text" name="email" />
    <input type="submit" value="Subscribe!" th:attr="value=#{subscribe.submit}"/>
  </fieldset>
</form>
```



```
<img src="../../images/gtvglogo.png"
     th:attr="src=@{/images/gtvglogo.png},title=#{logo},alt=#{logo}" />
```

设置img的 src、title、alt属性

6.2 基础语法

设置特定属性的值

```
<input type="submit" value="Subscribe!" th:value="#{subscribe.submit}"/>
```

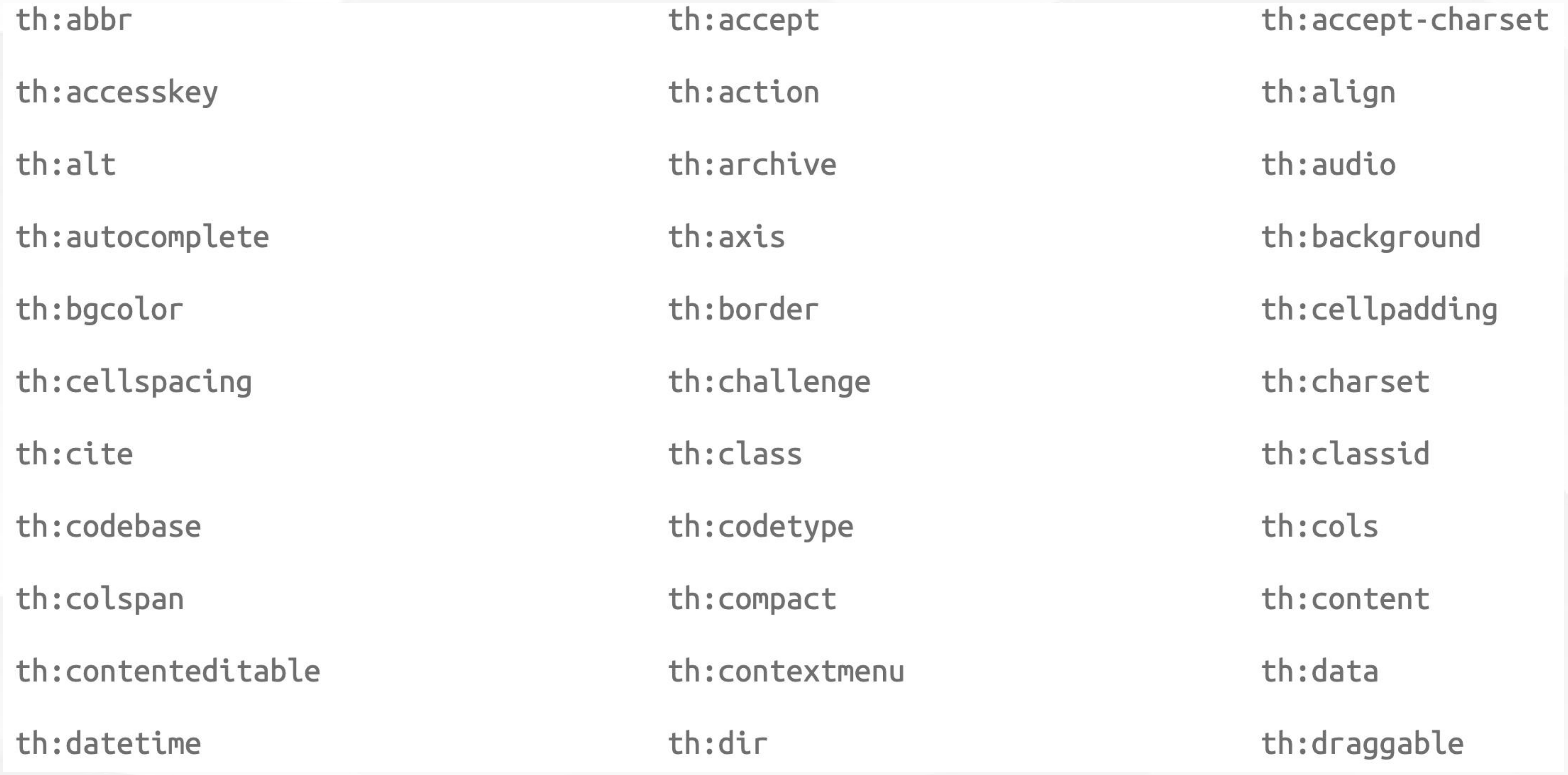
```
<form action="subscribe.html" th:action="@{/subscribe}">
```

```
<li><a href="product/list.html" th:href="@{/product/list}">Product List</a></li>
```

6.2 基础语法

设置特定属性的值

https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#setting-value-to-specific-attributes

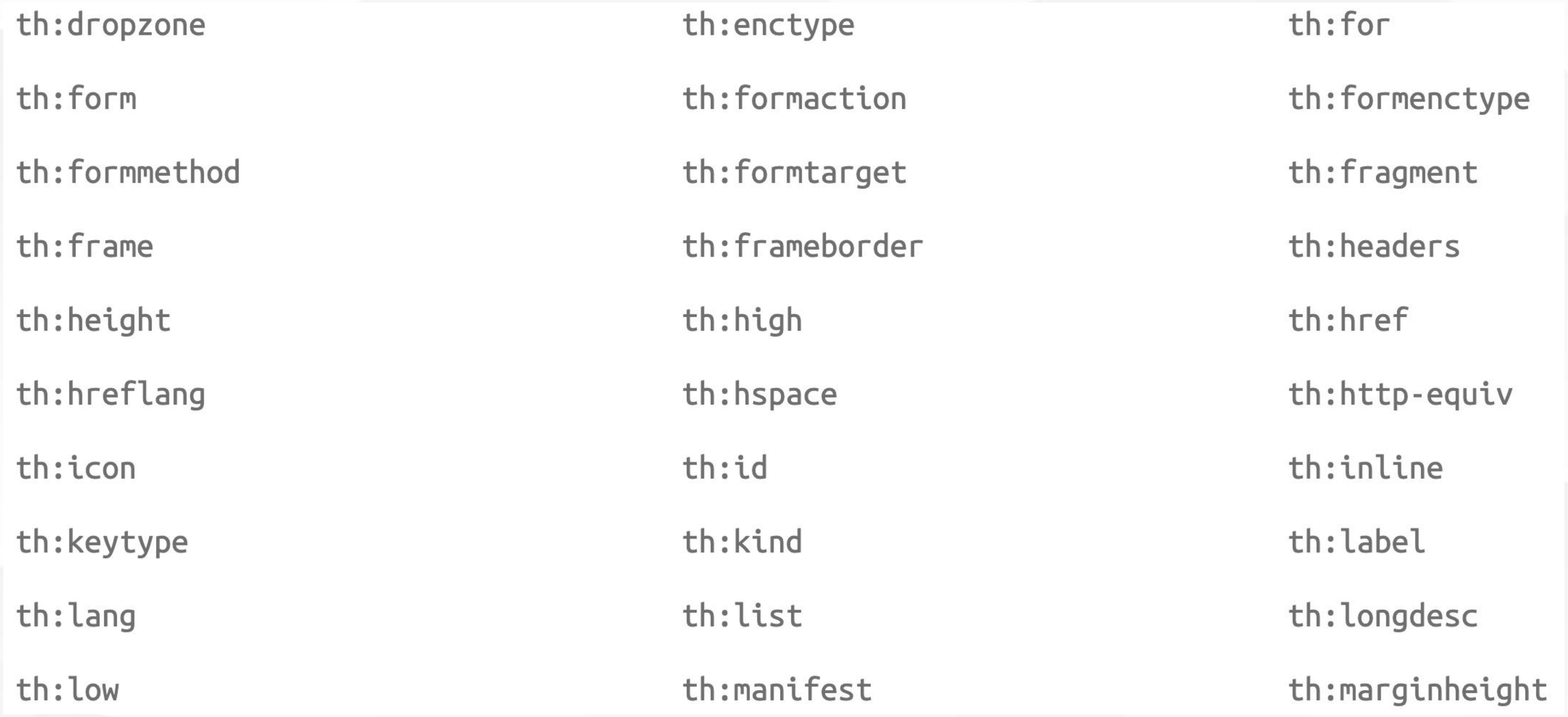


th:abbr	th:accept	th:accept-charset
th:accesskey	th:action	th:align
th:alt	th:archive	th:audio
th:autocomplete	th:axis	th:background
th:bgcolor	th:border	th:cellpadding
th:cellspacing	th:challenge	th:charset
th:cite	th:class	th:classid
th:codebase	th:codetype	th:cols
th:colspan	th:compact	th:content
th:contenteditable	th:contextmenu	th:data
th:datetime	th:dir	th:draggable

6.2 基础语法

设置特定属性的值

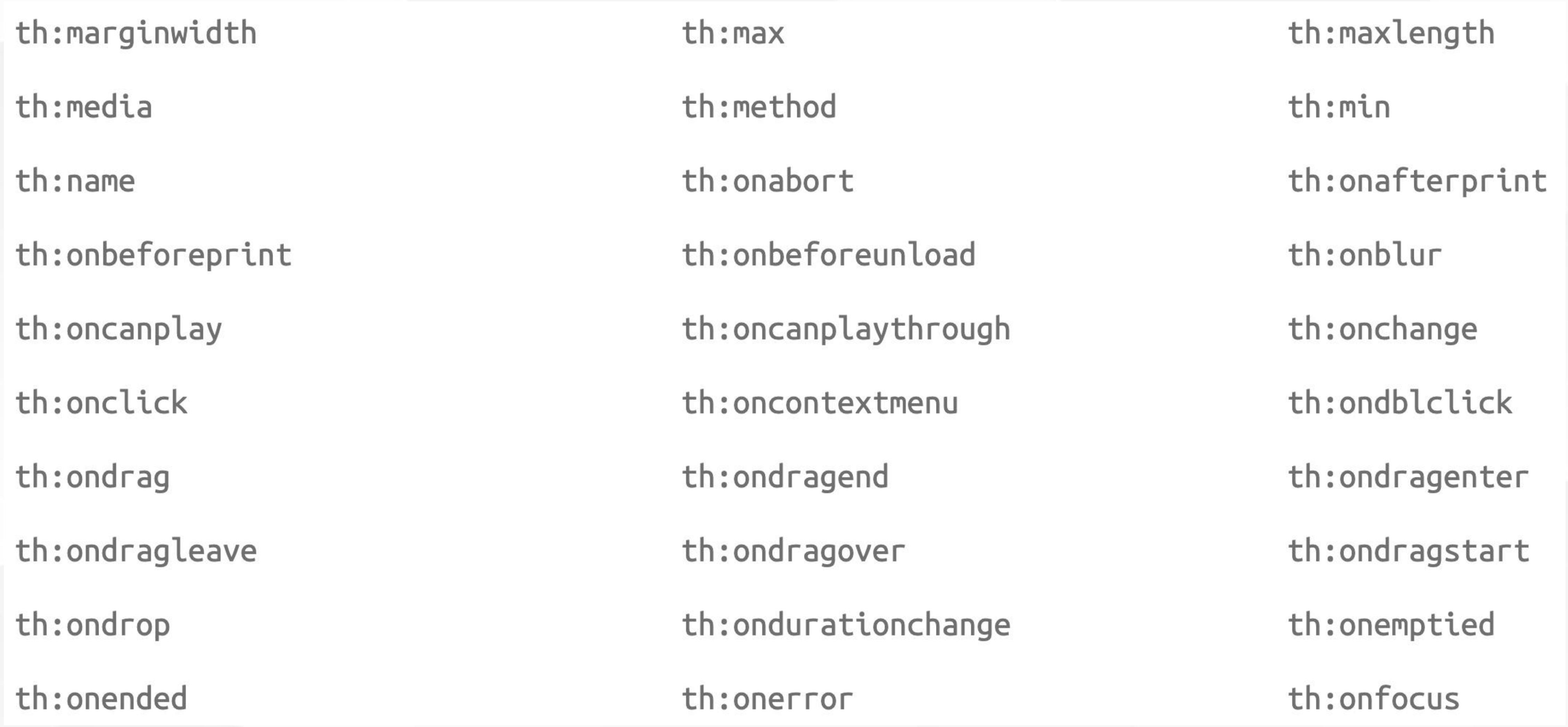
https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#setting-value-to-specific-attributes



th:dropzone	th:enctype	th:for
th:form	th:formaction	th:formenctype
th:formmethod	th:formtarget	th:fragment
th:frame	th:frameborder	th:headers
th:height	th:high	th:href
th:hreflang	th:hspace	th:http-equiv
th:icon	th:id	th:inline
th:keytype	th:kind	th:label
th:lang	th:list	th:longdesc
th:low	th:manifest	th:marginheight

6.2 基础语法

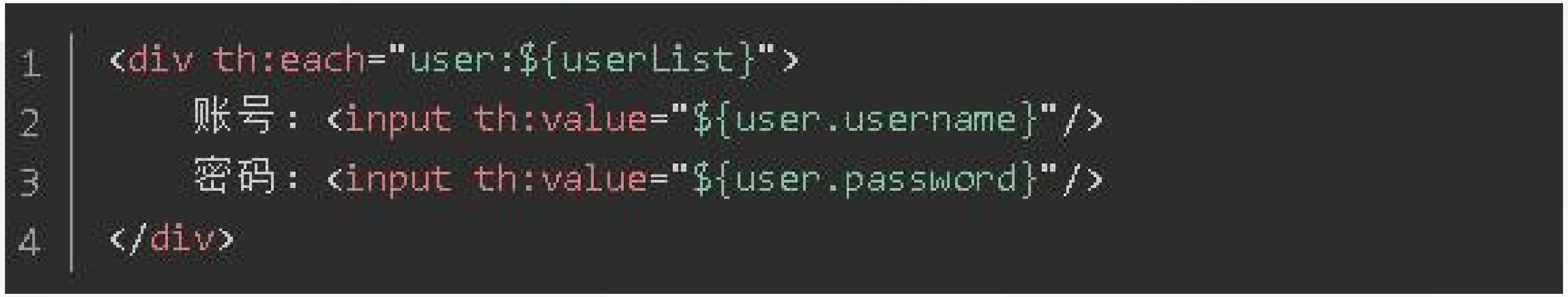
设置特定属性的值 https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html#setting-value-to-specific-attributes



th:marginwidth	th:max	th:maxlength
th:media	th:method	th:min
th:name	th:onabort	th:onafterprint
th:onbeforeprint	th:onbeforeunload	th:onblur
th:oncanplay	th:oncanplaythrough	th:onchange
th:onclick	th:oncontextmenu	th:ondblclick
th:ondrag	th:ondragend	th:ondragenter
th:ondragleave	th:ondragover	th:ondragstart
th:ondrop	th:ondurationchange	th:onemptied
th:onended	th:onerror	th:onfocus

6.3 迭代循环

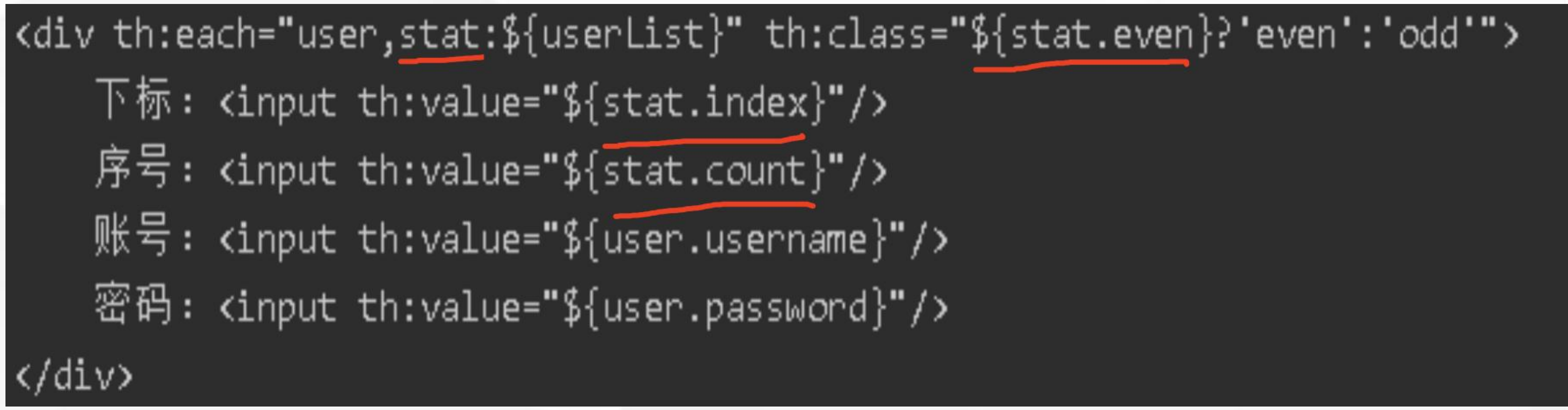
想要遍历List集合很简单，配合th:each 即可快速完成迭代。例如遍历用户列表：



```
<div th:each="user:${userList}">
    账号：<input th:value="${user.username}"/>
    密码：<input th:value="${user.password}"/>
</div>
```

在集合的迭代过程还可以获取状态变量，只需在变量后面指定状态变量名即可，状态变量可用于获取集合的下标/序号、总数、是否为单数/偶数行、是否为第一个/最后一个。例如：

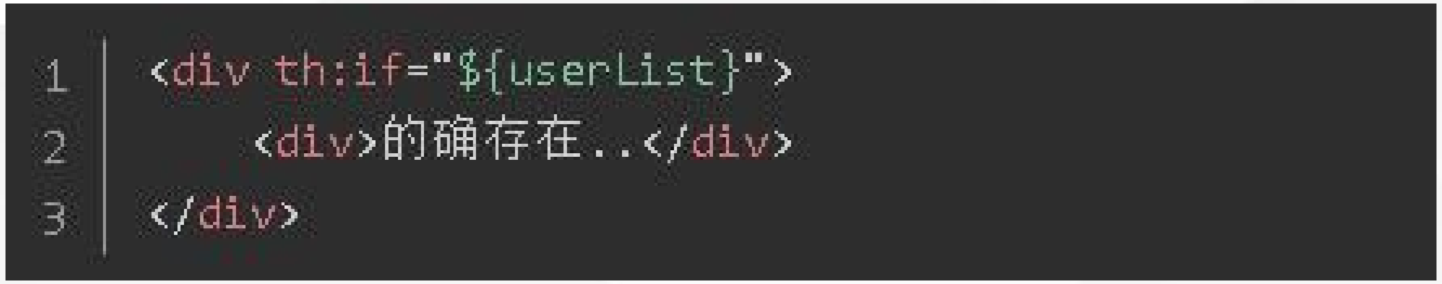
*index*属性：当前迭代索引，从0开始
*count*属性：当前迭代索引，从1开始
*size*属性：迭代变量中元素的总数
*current*属性：每次迭代的iter变量
*even/odd*属性：当前迭代是偶数还是奇数
*first*属性：当前迭代是否是第一次迭代。
*last*属性：当前迭代是否是最后一次迭代。



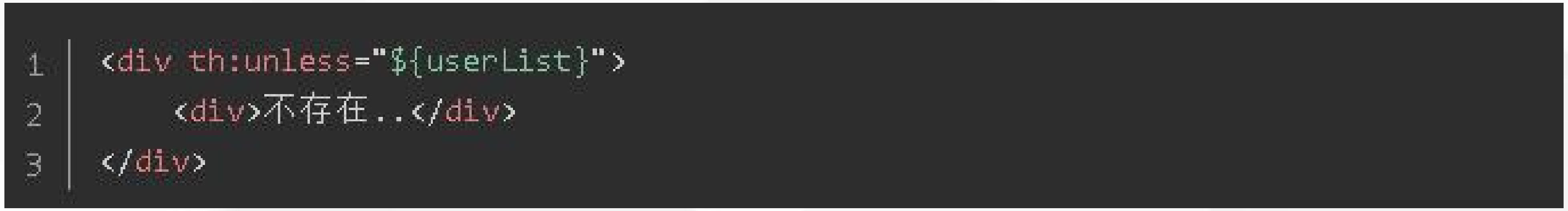
```
<div th:each="user,stat:${userList}" th:class="${stat.even}?'even':'odd'">
    下标：<input th:value="${stat.index}"/>
    序号：<input th:value="${stat.count}"/>
    账号：<input th:value="${user.username}"/>
    密码：<input th:value="${user.password}"/>
</div>
```

6.4 条件判断

使用th:if 和 th: unless进行条件判断



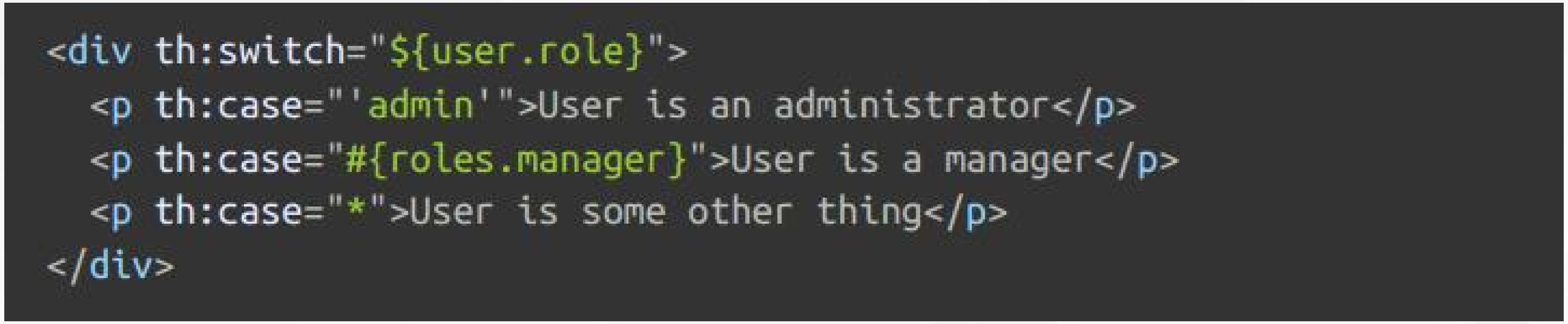
```
<div th:if="${userList}">
    <div>的确存在..</div>
</div>
```



```
<div th:unless="${userList}">
    <div>不存在..</div>
</div>
```

6.4 条件判断与分支

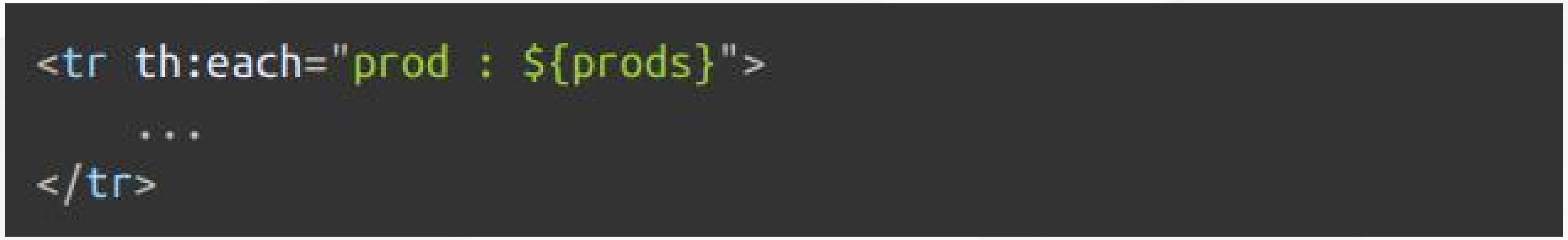
使用th:switch和 th: case进行分支



```
<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>
```

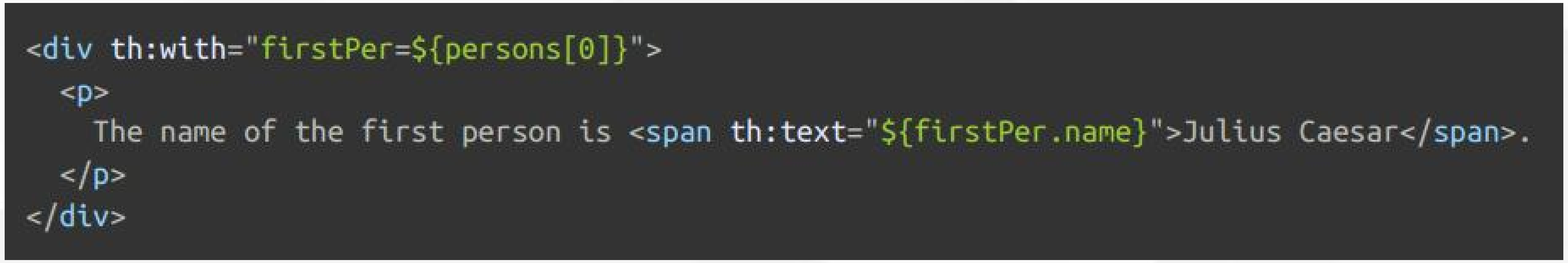
6.5.1 局部变量

prod只在<tr>标签内有效



```
<tr th:each="prod : ${prods}">
    ...
</tr>
```

使用th:with定义一个局部变量



```
<div th:with="firstPer=${persons[0]}">
  <p>
    The name of the first person is <span th:text="${firstPer.name}">Julius Caesar</span>.
  </p>
</div>
```

6.5.2 内联写法

内联写法：不使用扩展属性，直接生成到html中， [[${xx}]] 或[(${xx})]

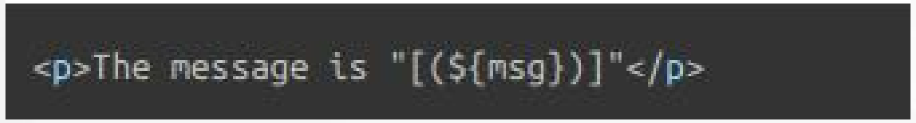
```
<p>Hello, [[${session.user.name}]]!</p>
```

等价于

```
<p>Hello, <span th:text="${session.user.name}">Sebastian</span>!</p>
```

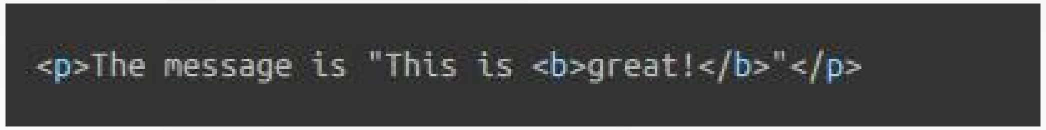
6.5.2 内联写法

内联写法： [[${xx}]] 或[(${xx})]，二者区别是前者为*HTML-escaped*， 后者不进行*HTML-escaped*，*后者类似于th:utext* 。例如当变量 msg= ｀This is <b>great</b> ｀时，

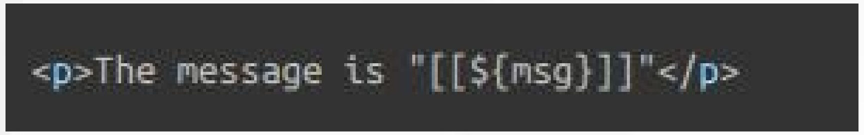


```
<p>The message is "[(${msg})]"</p>
```

不转义

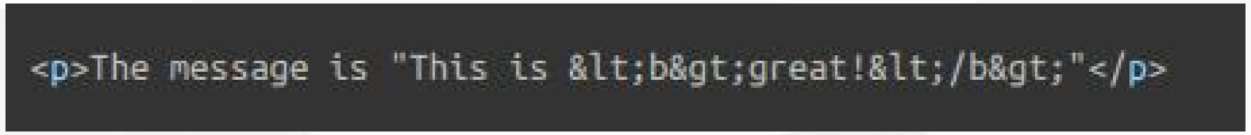


```
<p>The message is "This is <b>great!</b>"</p>
```

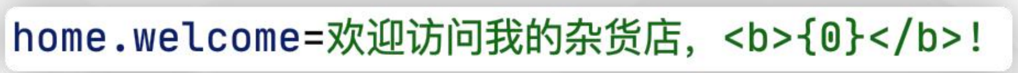


```
<p>The message is "[[${msg}]]"</p>
```

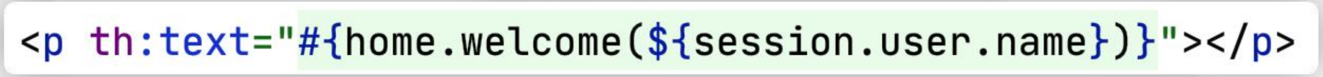
转义



```
<p>The message is "This is &lt;b&gt;great!&lt;/b&gt;"</p>
```

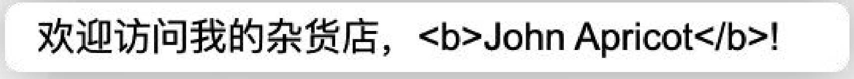


```
home.welcome=欢迎访问我的杂货店，<b>{0}</b>!
```

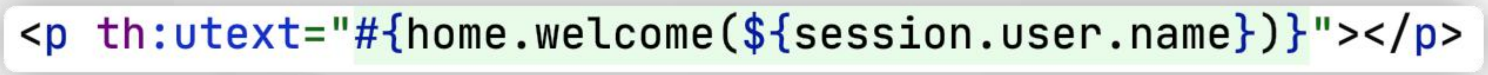


```
<p th:text="#{home.welcome(${session.user.name})}"></p>
```

转义

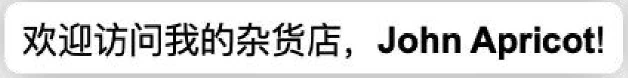


欢迎访问我的杂货店，<b>John Apricot</b>!



```
<p th:utext="#{home.welcome(${session.user.name})}"></p>
```

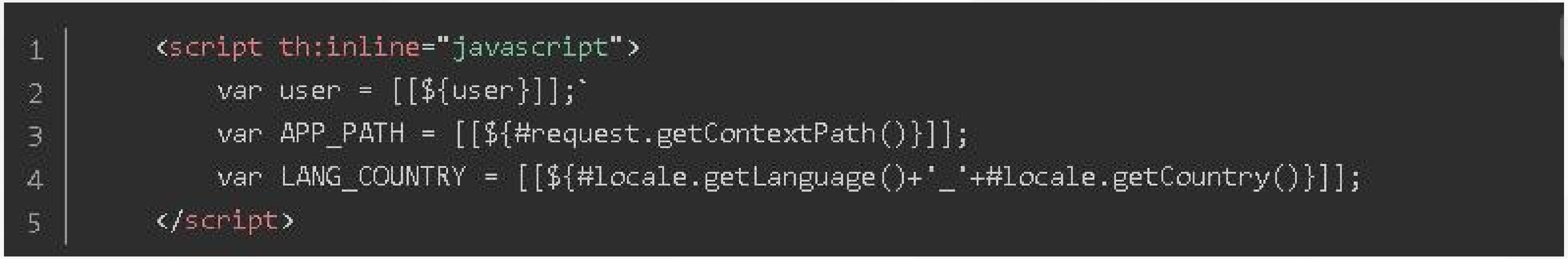
不转义



欢迎访问我的杂货店，**John Apricot!**

6.5.2 内联写法

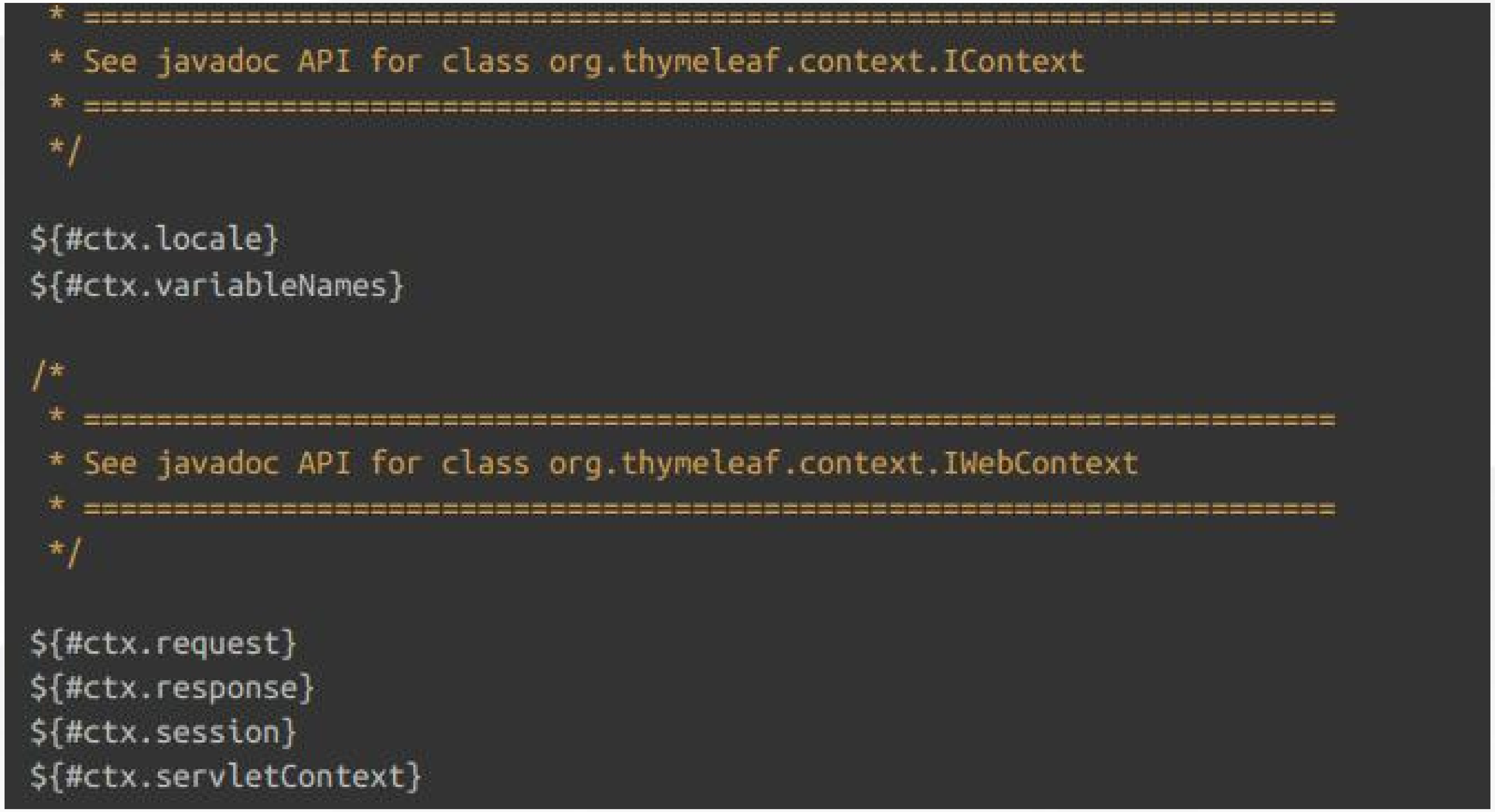
JavaScript内联，th:inline=“javascript”



```
<script th:inline="javascript">
    var user = [[${user}]];`
    var APP_PATH = [[${#request.getContextPath()}]];
    var LANG_COUNTRY = [[${#locale.getLanguage()+'_'+#locale.getCountry()}]];
</script>
```

6.5.3 内置对象(简要了解)

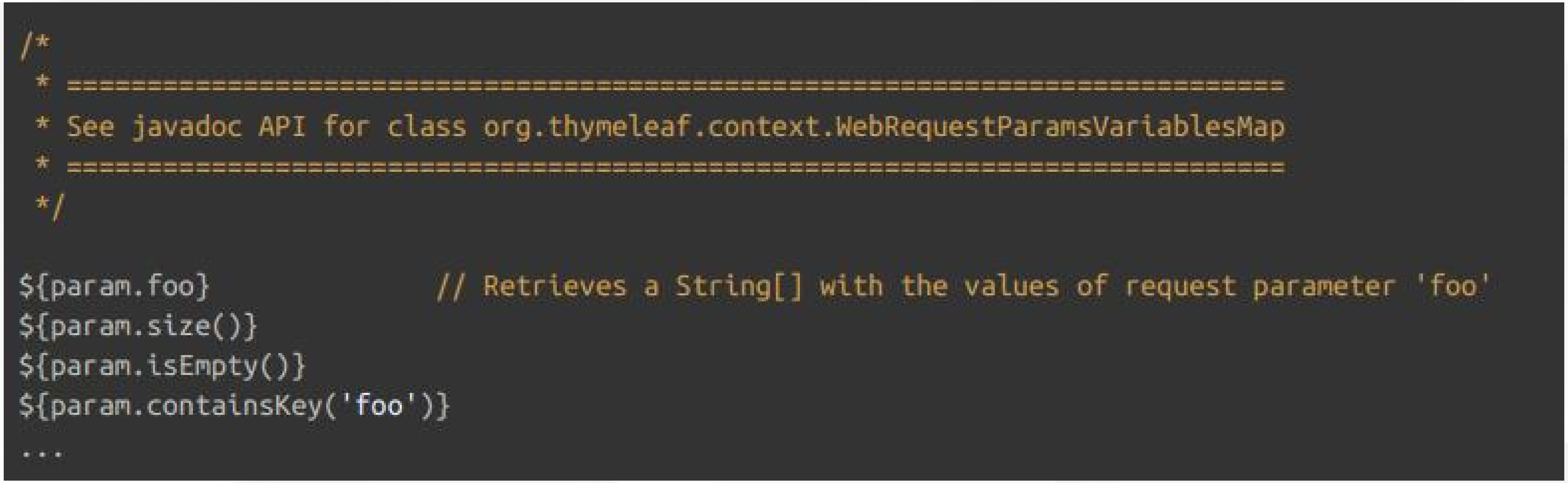
${#ctx} 上下文对象，可用于获取其它内置对象。



```
 * =======================================================================
 * See javadoc API for class org.thymeleaf.context.IContext
 * =======================================================================
 */

${#ctx.locale}
${#ctx.variableNames}

/*
 * =======================================================================
 * See javadoc API for class org.thymeleaf.context.IWebContext
 * =======================================================================
 */

${#ctx.request}
${#ctx.response}
${#ctx.session}
${#ctx.servletContext}
```

6.5.3 内置对象(简要了解)

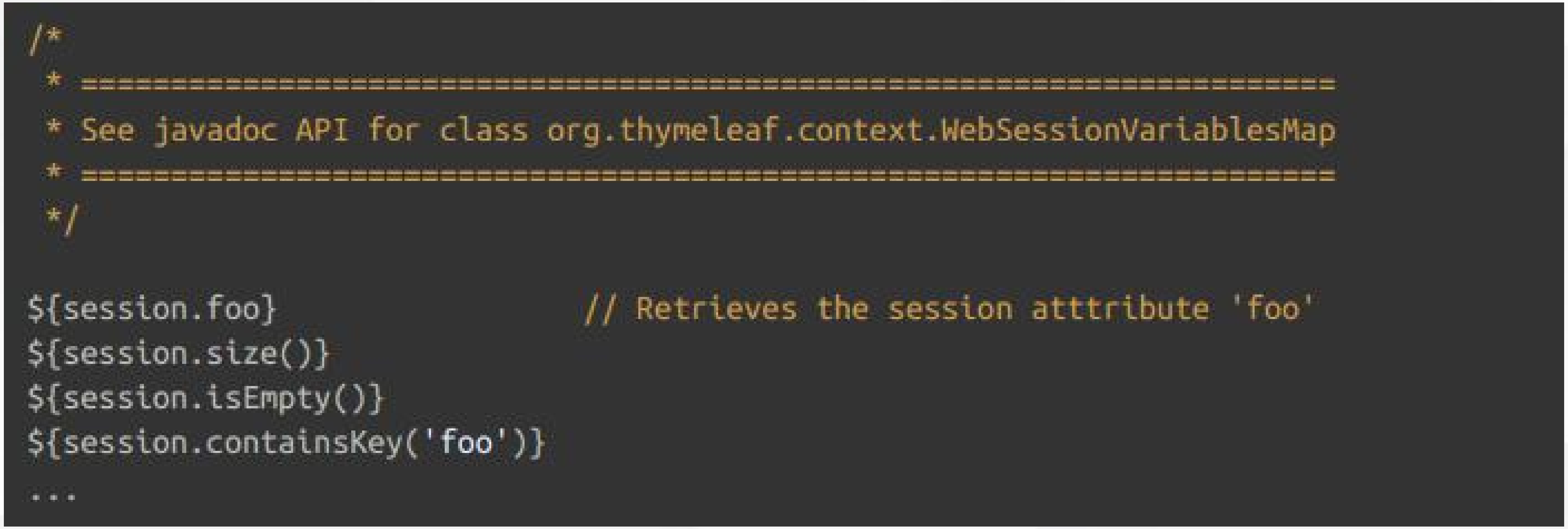
${param }：获取请求参数。



```
/*
 * ============================================================================
 * See javadoc API for class org.thymeleaf.context.WebRequestParamsVariablesMap
 * ============================================================================
 */

${param.foo}              // Retrieves a String[] with the values of request parameter 'foo'
${param.size()}
${param.isEmpty()}
${param.containsKey('foo')}
...
```

6.5.3 内置对象(简要了解)

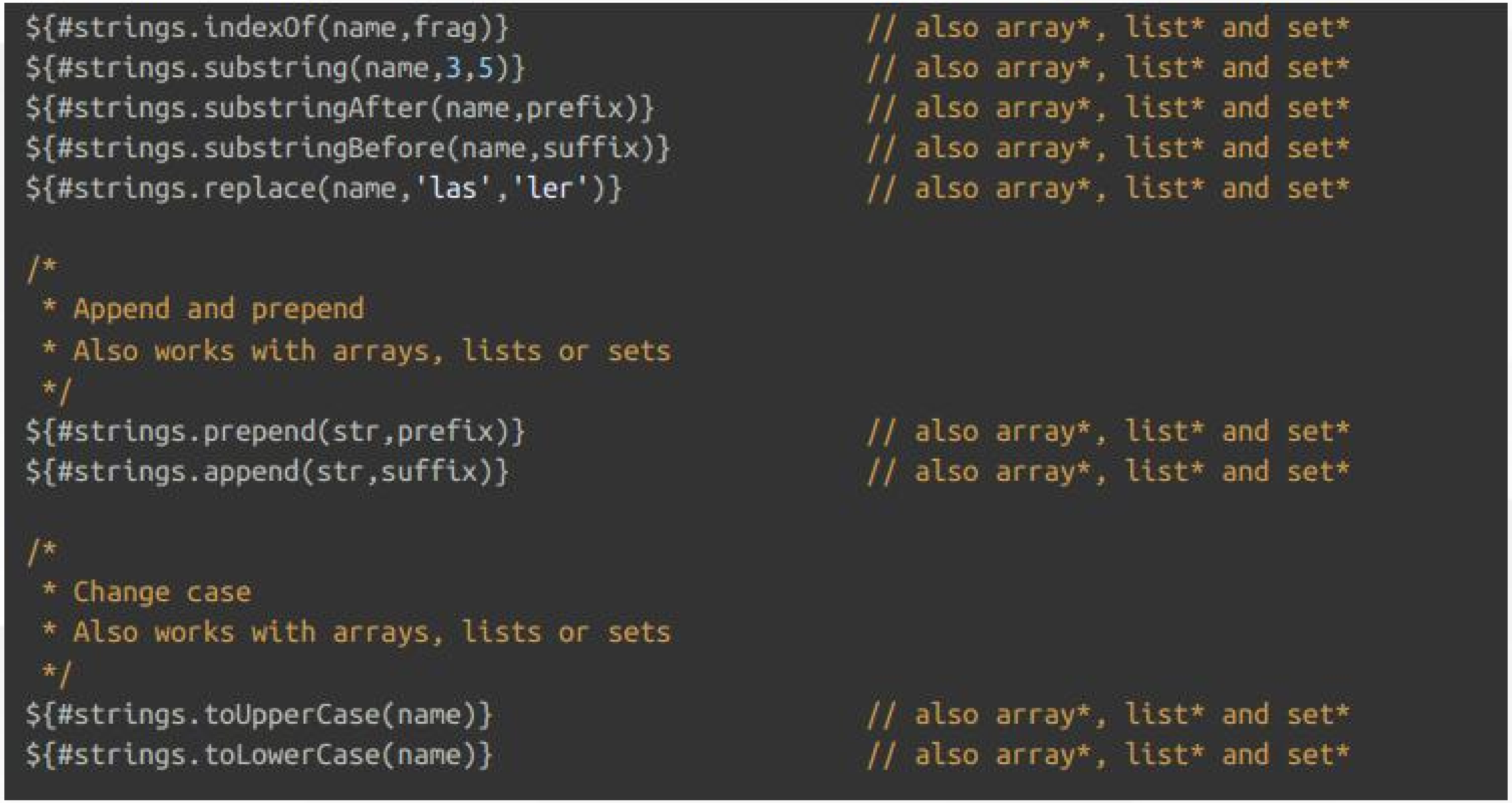
${session }：获取session属性。



```
/*
 * ============================================================================
 * See javadoc API for class org.thymeleaf.context.WebSessionVariablesMap
 * ============================================================================
 */

${session.foo}                  // Retrieves the session atttribute 'foo'
${session.size()}
${session.isEmpty()}
${session.containsKey('foo')}
...
```

6.5.3 内置对象(简要了解)

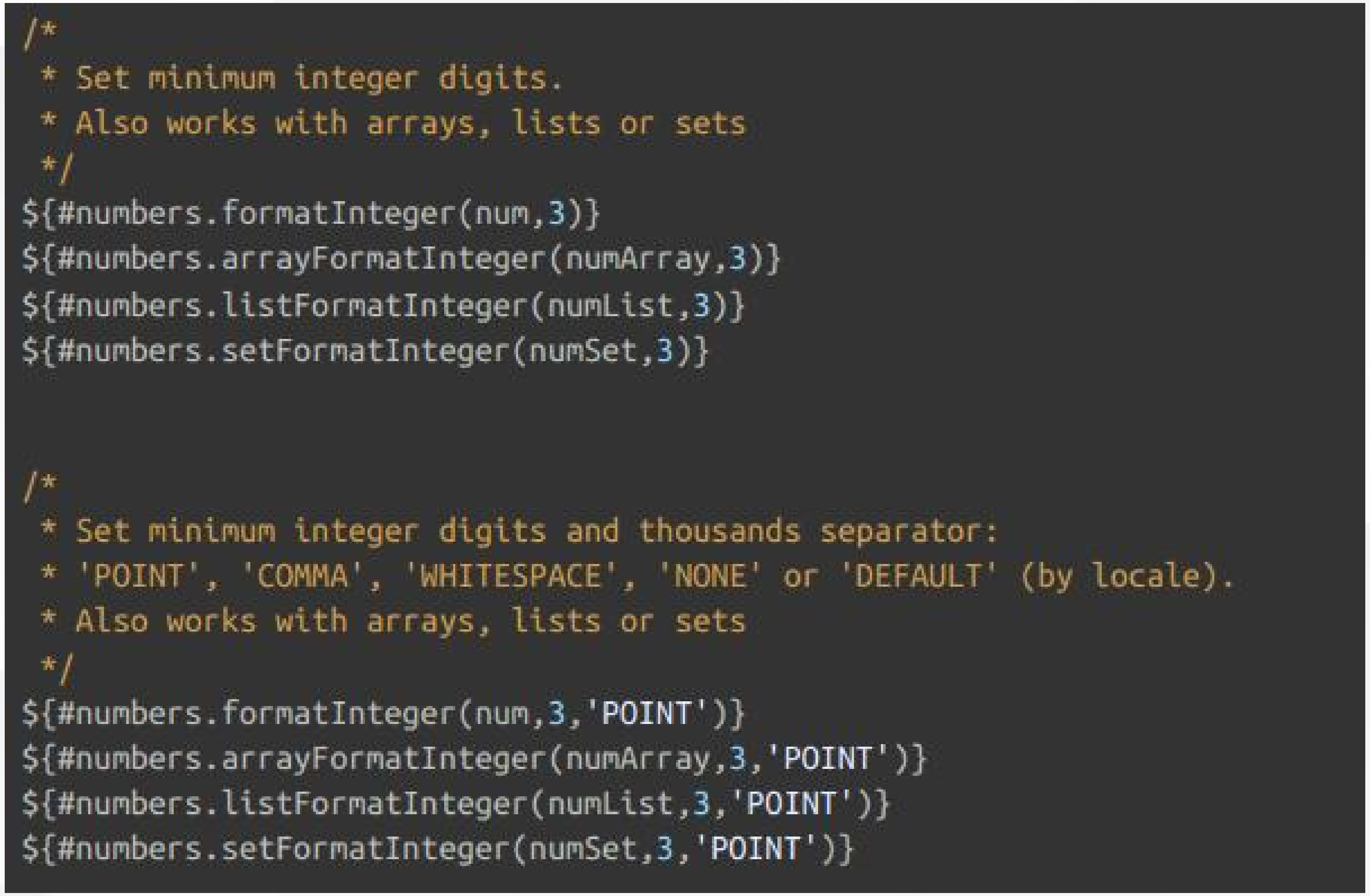
${#strings }：strings工具类。



```
${#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*
```

6.5.3 内置对象(简要了解)

${#numbers **}：numbers工具类。**



```
/*
 * 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')}
```

6.5.3 内置对象(简要了解)

#lists：List 工具类

#arrays：数组工具类

#sets：Set 工具类

#maps：常用Map方法。

#objects：一般对象类，通常用来判断非空

#bools：常用的布尔方法。

6.5.3 内置对象(简要了解)

#execInfo：获取页面模板的处理信息。

#messages：在变量表达式中获取外部消息的方法，与使用＃{...}语法获取的方法相同。

#uris：转义部分URL / URI的方法。

#conversions：用于执行已配置的转换服务的方法。

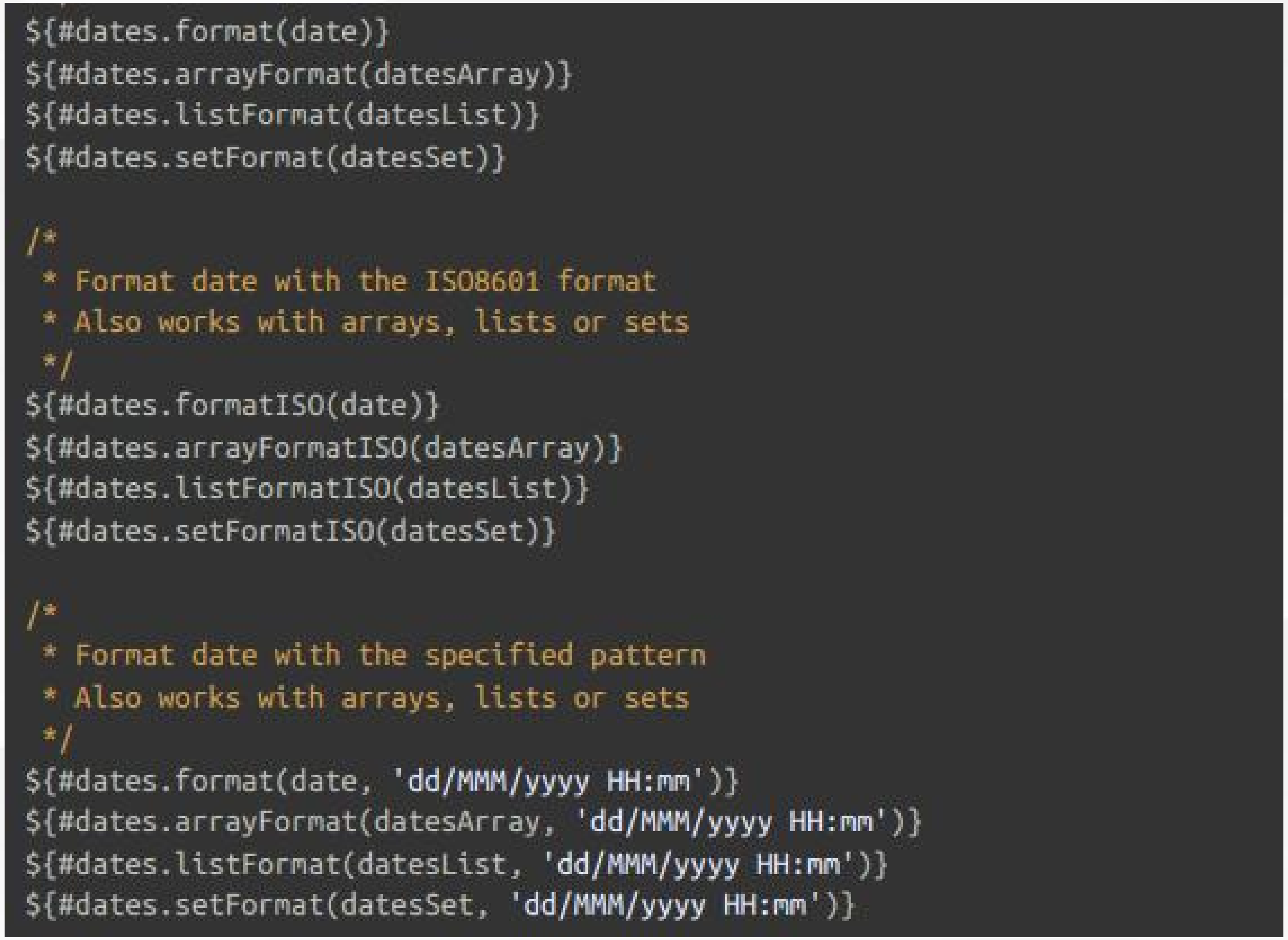
#dates：时间操作和时间格式化等。

#calendars：用于更复杂时间的格式化。

#aggregates：在数组或集合上创建聚合的方法。

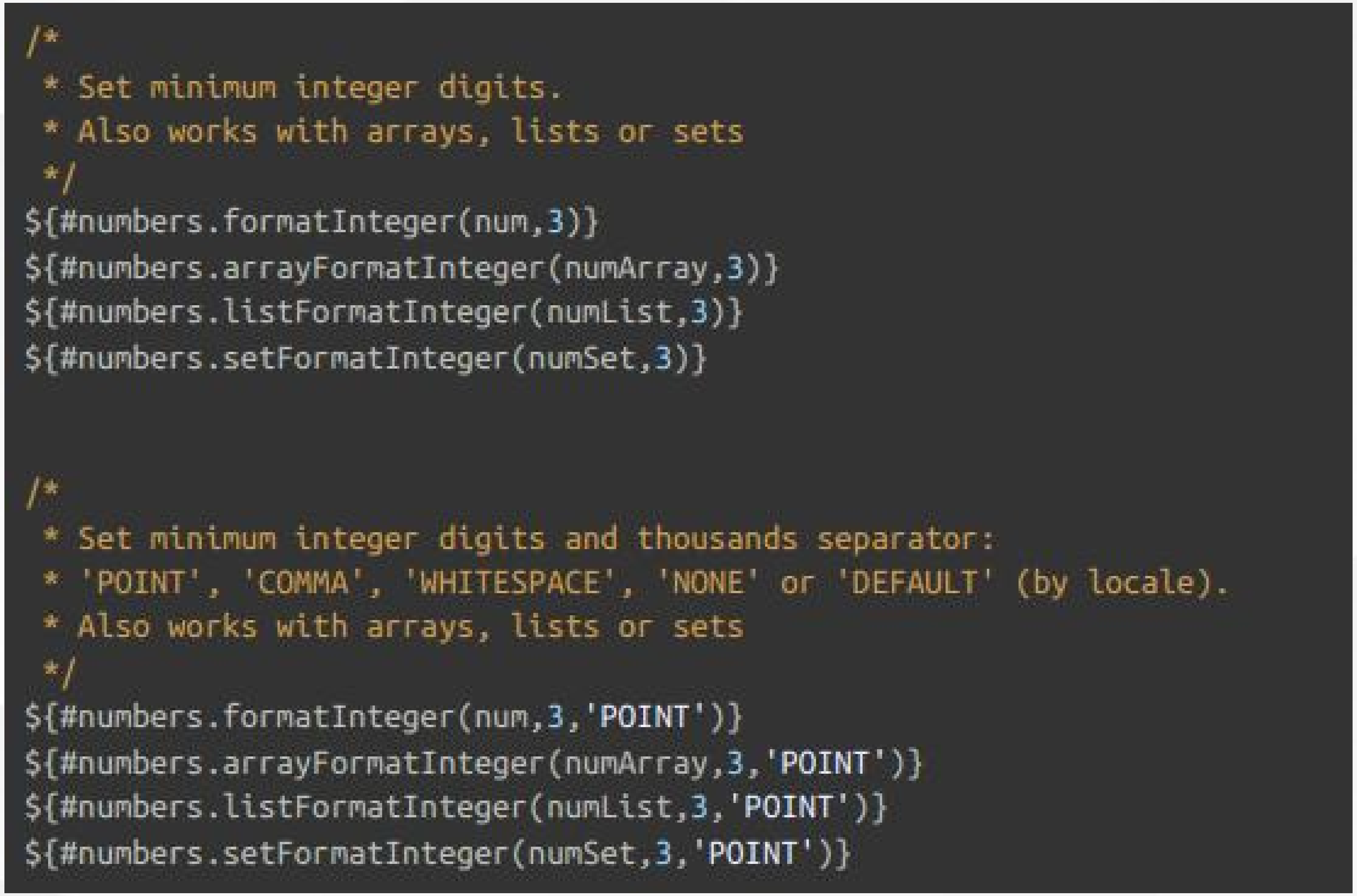
#ids：处理可能重复的id属性的方法。

6.5.3 内置对象(简要了解)



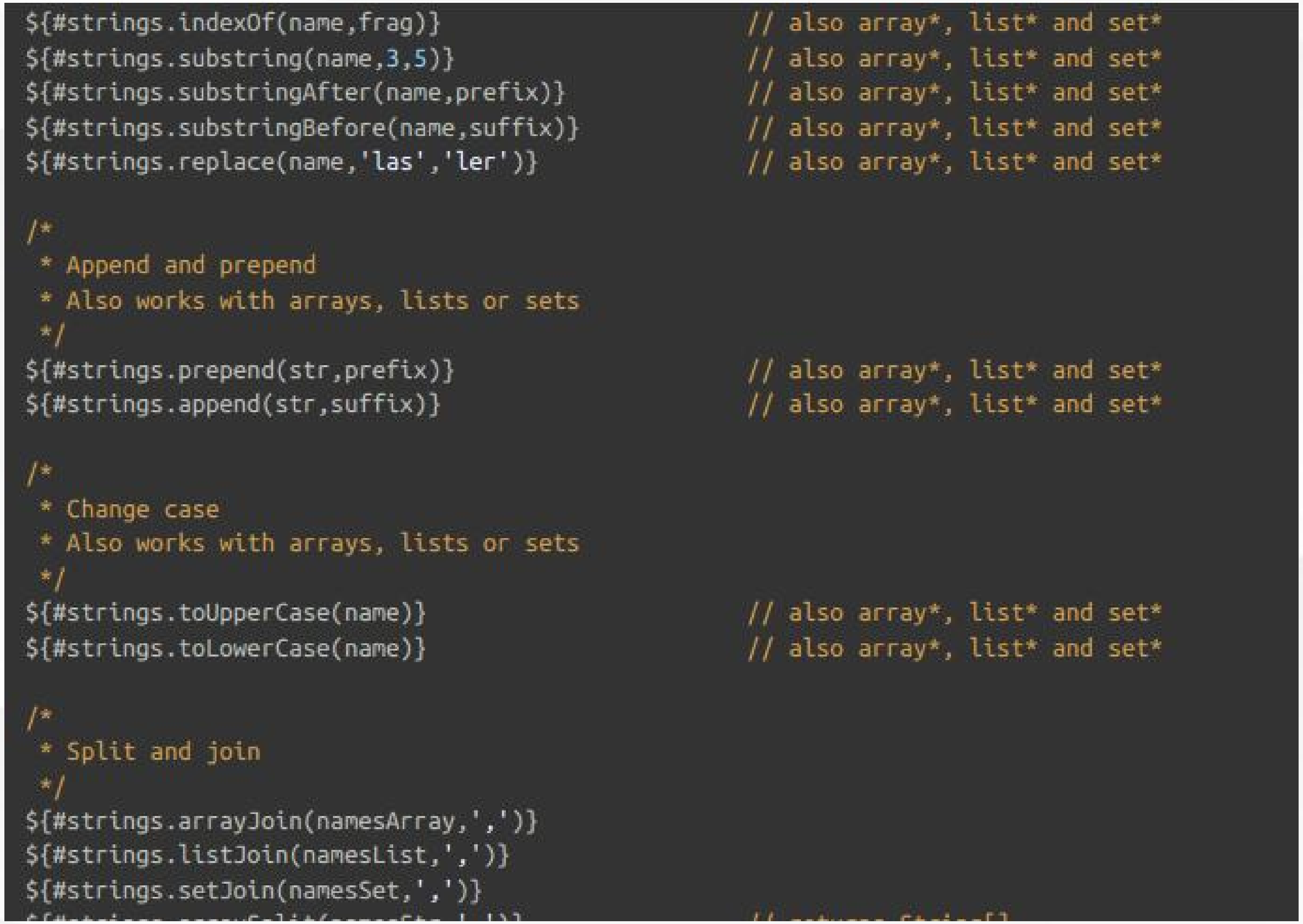
```
${#dates.format(date)}
${#dates.arrayFormat(datesArray)}
${#dates.listFormat(datesList)}
${#dates.setFormat(datesSet)}

/*
 * 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')}
```

6.5.3 内置对象(简要了解)

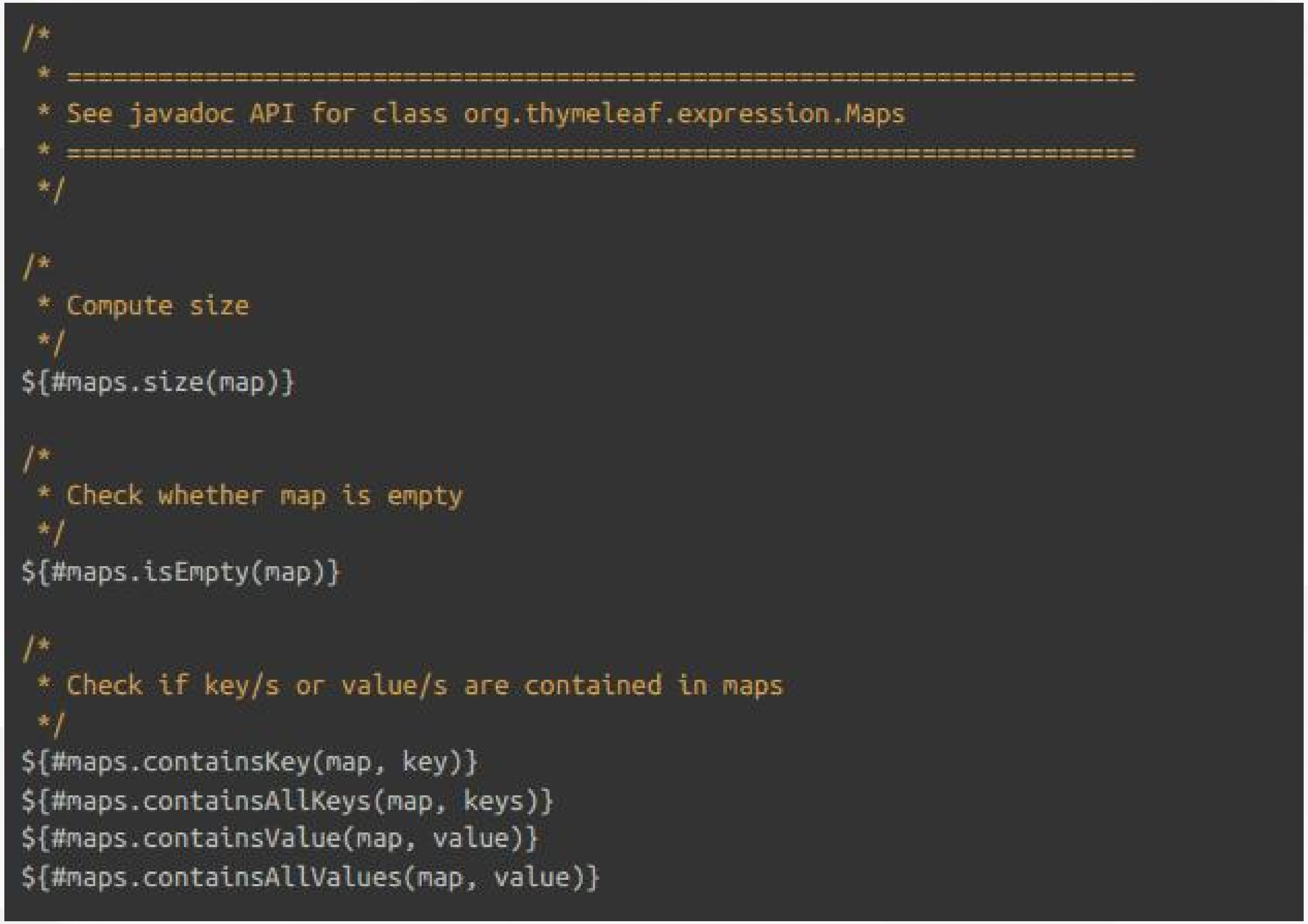


```
/*
 * Set minimum integer digits.
 * Also works with arrays, lists or sets
 */
${#numbers.formatInteger(num,3)}
${#numbers.arrayFormatInteger(numArray,3)}
${#numbers.listFormatInteger(numList,3)}
${#numbers.setFormatInteger(numSet,3)}


/*
 * 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')}
```

6.5.3 内置对象(简要了解)


```
${#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[]
```

6.5.3 内置对象(简要了解)



```
/*
 * ======================================================================
 * See javadoc API for class org.thymeleaf.expression.Maps
 * ======================================================================
 */

/*
 * Compute size
 */
${#maps.size(map)}

/*
 * Check whether map is empty
 */
${#maps.isEmpty(map)}

/*
 * Check if key/s or value/s are contained in maps
 */
${#maps.containsKey(map, key)}
${#maps.containsAllKeys(map, keys)}
${#maps.containsValue(map, value)}
${#maps.containsAllValues(map, value)}
```