

Thymeleaf

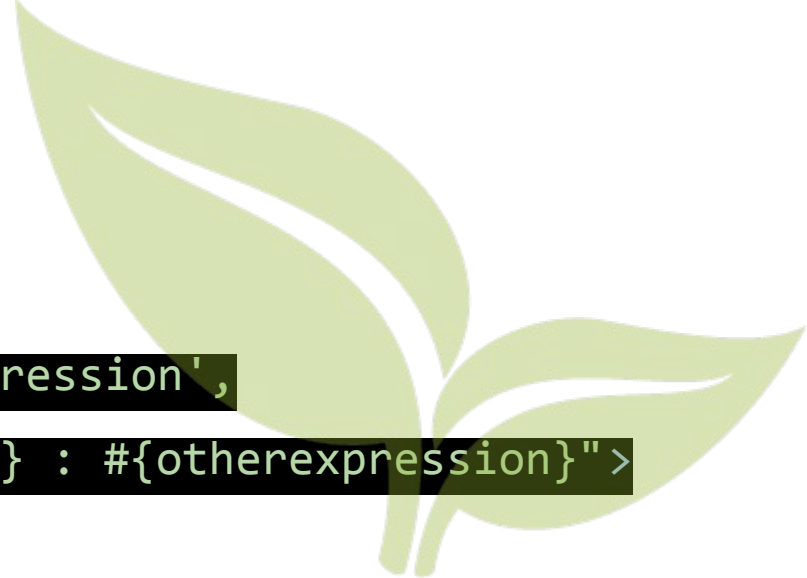


Kristina Sukhomlina
A4M35OSP

Thymeleaf: assignment

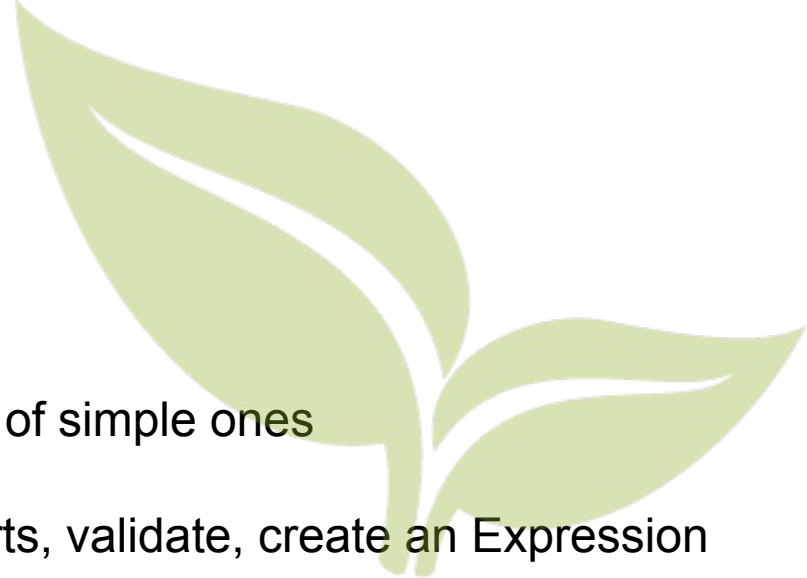
```
<div th:class="${someVariable} ??  
    'value' : 'expression',  
    ${anotherValue} : #{otherexpression}">
```

- Add a switch-case complex expression to the library
- Enable its parsing and executing



Thymeleaf: solution

- each Complex expression is a container full of simple ones
- *Composition*: parse the input, divide into parts, validate, create an Expression
- *Execution*: imitate the 'switch case' behavior



Thymeleaf: solution

- *Composition:*

Parsing node → String input → SwitchCaseExpression

Switch Case expression: condition expr + case sequence

- *Execution:*

Execute complex → Execute switch/case → Match condition to best case



Code screenshots

thymeleaf#112 (comment)

commit: added Switch Case expression and its helping classes

3.0-dev (#496)

sukhokri committed 2 hours ago

1 parent 8f6bbae commit 674dab0b4f507f2c

Showing 6 changed files with 468 additions and 0 deletions.

81 src/main/java/org/thymeleaf/standard/expression/Case.java

```
object executeSwitchCase(IExpressionContext context, SwitchCaseExpression expression,
    StandardExpressionExecutionContext expContext) {
    if (logger.isTraceEnabled()) {
        logger.trace("[THYMELEAF][{}] Evaluating switch expression: \"{}\\\"", TemplateEngine.threadIndex(),
            expression.getStringRepresentation());
    }

```

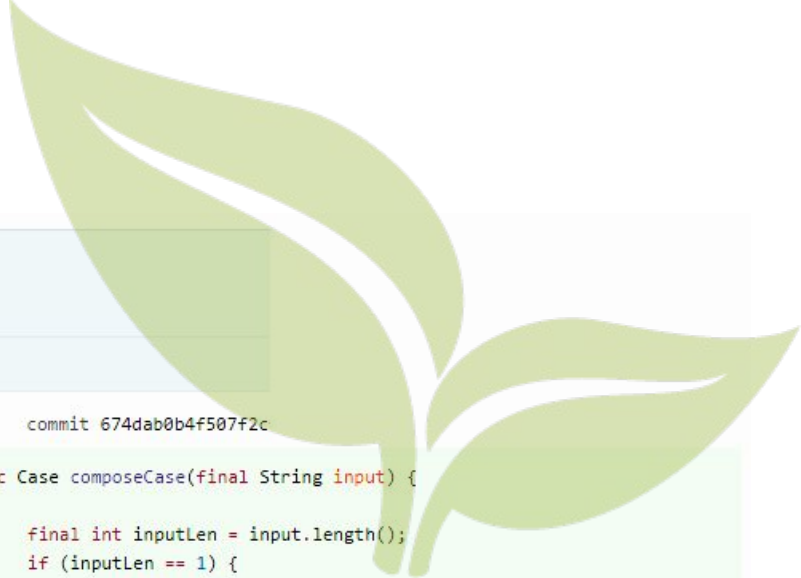
in switch condition

```
object switchObj = expression.getSwitchConditionExpression().execute(context, expContext);
defaultCase = null;
```

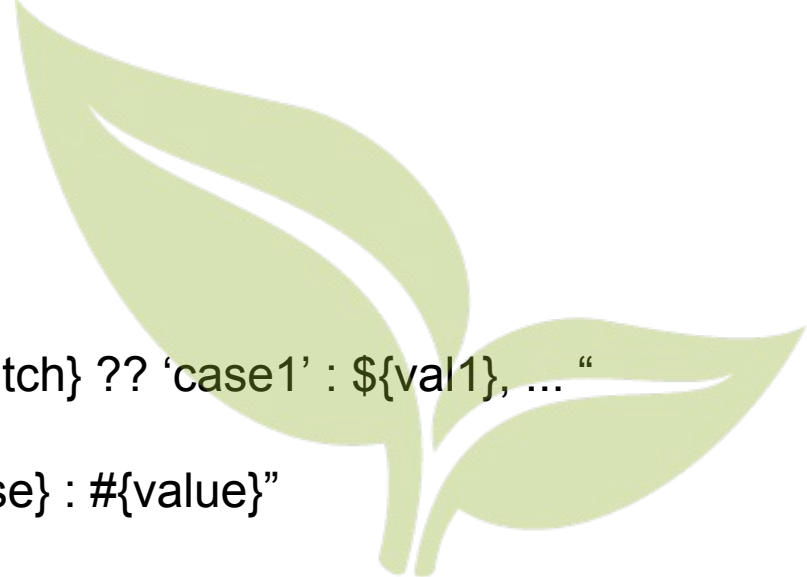
are each case 'key' with switch variable

the switch variable matches on of the cases, return its value

```
86 +
87 +     static Case composeCase(final String input) {
88 +
89 +         final int inputLen = input.length();
90 +         if (inputLen == 1) {
91 +             return null;
92 +         }
93 +         final int operatorPos = input.indexOf(':');
94 +
95 +         final String leftInput = input.substring(0, operatorPos).trim();
96 +         final String rightInput = input.substring(operatorPos + 1).trim();
97 +
98 +         if (StringUtils.isEmptyOrWhitespace(leftInput)) {
99 +             return null;
100 +         }
101 +
102 +         final Expression leftExpr = ExpressionParsingUtil.parseAndCompose(Expression leftInput);
103 +         if (leftExpr == null) {
104 +             return null;
105 +         }
106 +     }
107 + }
```



Thymeleaf: technical details



- SwitchCaseExpression.java // “\${switch} ?? ‘case1’ : \${val1}, ... “
- Case.java // “\${case} : #{value}”
- CaseUtils.java // composes sequence and cases
- CaseSequence.java // reads “case1,case2 ...”
- added calls to existing services // composition and execution calls

Thymeleaf: community



- changes available in public repository
- Pull request
- fast response to contribution request, waiting for changes approval

Thymeleaf

Thank you for attention

<http://www.thymeleaf.org/>

