



FPT POLYTECHNIC



www.poly.edu.vn



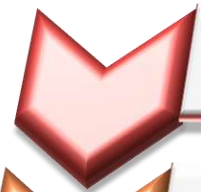
Conceive Design Implement Operate

BASIC THYMELEAF

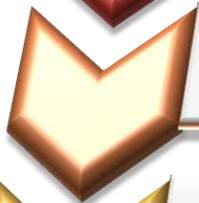
GIẢNG VIÊN: NGUYỄN NGHIÊM



<https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html>



- Spring Boot & JSP ViewResolver



- Thymeleaf Introduction



- Standard Expressions



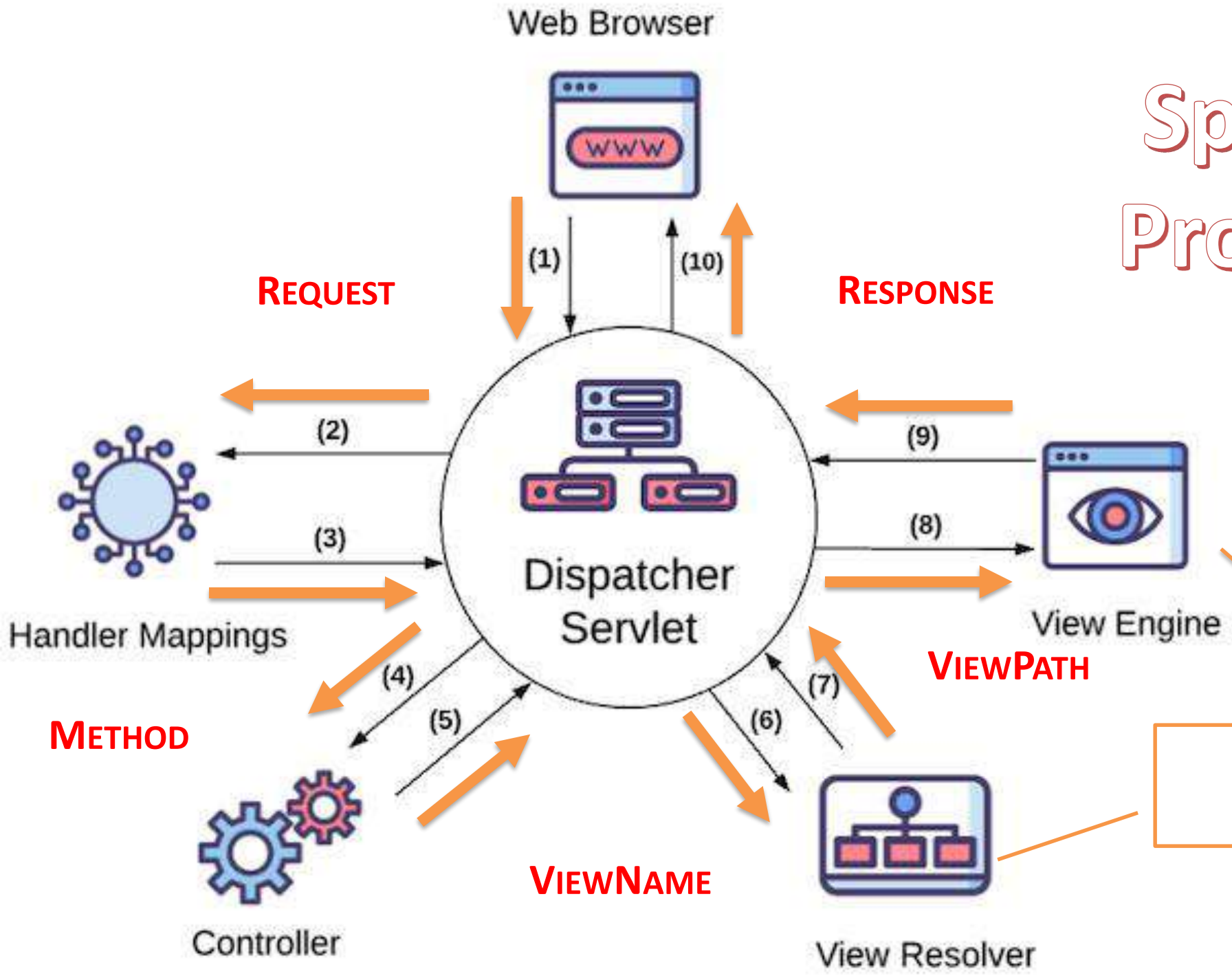
- Thymeleaf Attributes



- Thymeleaf Utilities



Spring MVC Process Flow



JSP, **THYMELEAF**...

- ❑ JSP, EL & JSTL
- ❑ Spring & Spring Form
- ❑ Tiles Framework

```
<%@ page pageEncoding="utf-8"%>
```

```
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
```

```
<%@ taglib uri="http://java.sun.com/jstl/fmt_rt" prefix="fmt" %>
```

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```

```
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
```

```
<%@ taglib uri="http://www.springframework.org/tags" prefix="spring" %>
```

```
<%@ taglib uri="http://tiles.apache.org/tags-tiles" prefix="tiles" %>
```

❑ Phức tạp

- ❖ Khai báo đúng thư viện thẻ cần thiết cho trang JSP (*.jsp)
- ❖ Khai báo thư viện phụ thuộc (pom.xml)
- ❖ Khai báo ViewResolver (application.properties)
- ❖ Cấu hình bean cho ViewResolver (@Configuration)

❑ Khó khăn

- ❖ Quá nhiều thứ để nhớ, để làm
- ❖ Mã phân tán, khó quản lý
- ❖ Tạo khoảng cách lớn giữa designer và developer

❑ *Chú ý: dù có đôi chút khó khăn nhưng JSP vẫn được sử dụng phổ biến nhất hiện nay trong các dự án Java Web.*



THYMELEAF INTRODUCTION

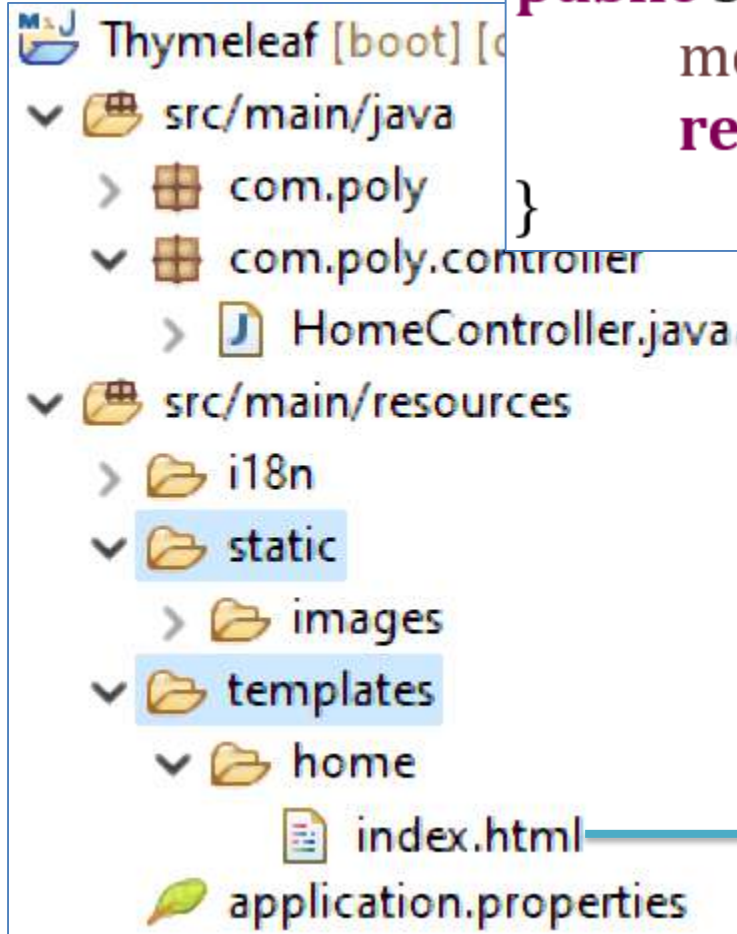
- ❑ Thymeleaf là một động cơ tạo mẫu (template engine) chạy phía server có khả năng xử lý việc tạo ra HTML, XML, CSS, JS và TEXT.
- ❑ Thymeleaf dựa vào các nguyên mẫu (HTML, CSS...) và bổ sung các quy luật cần thiết để sản sinh ra template hoàn thiện với dữ liệu động.
- ❑ Ưu điểm:
 - ❖ Đơn giản, dễ quản lý, bảo trì
 - ❖ Thu hẹp khoảng cách giữa designer và developer

```
<dependency>
```

```
    <groupId>org.springframework.boot</groupId>
```

```
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
```

```
</dependency>
```



```
@RequestMapping("/home/index")  
public String index(Model model) {  
    model.addAttribute("message", "FPT <b>Polytechnic</b>");  
    return "home/index";  
}
```

```
<!DOCTYPE html>  
<html xmlns:th="http://www.thymeleaf.org">  
<head>  
    <meta charset="utf-8">  
    <title>Insert title here</title>  
</head>  
<body>  
    <span th:text="${message}"></span>  
</body>  
</html>
```


❑ Thymeleaf namespace

❖ `<html xmlns:th="http://www.thymeleaf.org">`

❑ **th:text**

❖ ` `

❖ ` [[${message}]] `

FPT **Polytechnic**

Generated HTML: `FPT Polytechnic`

❑ **th:utext** (unescape text)

❖ ` `

❖ ` [(${message})] `

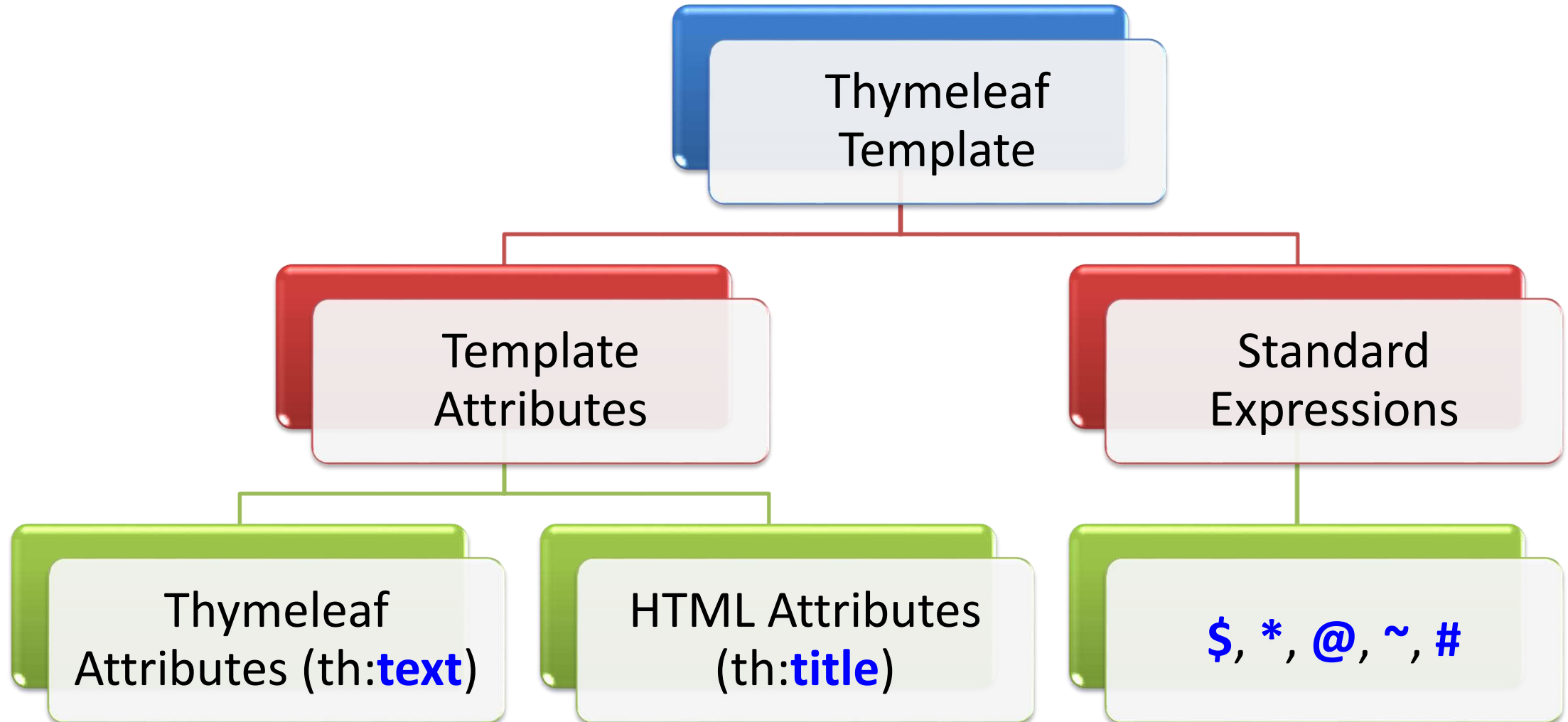
FPT Polytechnic

Generated HTML: `FPT Polytechnic `

DEMO



th:attribute = “Standard Expression”



THYMELEAF ATTRIBUTE & PRECEEDING

TT	HÀNH ĐỘNG	CÁC THUỘC TÍNH
1	Bao hàm các fragment	th: insert , th: replace
2	Vòng lặp	th: each
3	Đánh giá điều kiện	th: if , th: unless , th: switch , th: case
4	Định nghĩa biến cục bộ	th: object , th: with
5	Hiệu chỉnh thuộc tính	th: attr , th: attrprepend , th: attrappend
6	Hiệu chỉnh thuộc tính HTML	th: value , th: href , th: src ...
7	Thay đổi nội dung thẻ	th: text , th: utext
8	Khai báo fragment	th: fragment
9	Xóa fragment	th: remove



- ❑ **`${...}`**: Variable expressions
 - ❖ Scope and parameters
- ❑ **`*{...}`**: Selection expressions
 - ❖ Thuộc tính của **`th:object`**
- ❑ **`#{...}`**: Message expressions
 - ❖ I18n resource (**`.properties`**)
- ❑ **`@{...}`**: URL expressions
 - ❖ URL/URI
- ❑ **`~{...}`**: Fragment expressions
 - ❖ Layout (**`th:insert`**, **`th:replace`**)

STANDARD EXPRESSION EXAMPLE

```

```

```
<th:block th:replace="~/layout/menu.html" />
```

```
<ul>
```

```
    <li th:utext="${message}"/>
```

```
    <li th:text="${bean.name}" />
```

```
</ul>
```

```
<ul th:object="${bean}">
```

```
    <li th:text="*{name}" />
```

```
</ul>
```

```
<ul>
```

```
    <li th:text="#{menu.home}" />
```

```
</ul>
```

☐ th:**src**="@{path}"

☐ <th:**block**>

☐ th:**replace**="~{fragment}"

☐ th:**text**="\${text/plain}"

☐ th:**utext**="\${text/html}"

☐ th:**object**="\${bean}"

☐ th:text="*{property}"

☐ th:text="#{resourceKey}"

DEMO





STANDARD EXPRESSIONS

- ❑ ``
 - ❖ Hiển thị thuộc tính name của bean user trong **model** (**request**)
- ❑ `<li th:each="book : ${books}" th:text="${book.title}">>`
 - ❖ Duyệt tập hợp books trong model và hiển thị title của mỗi book
- ❑ `>`
 - ❖ Hiển thị thuộc tính name của bean user trong session scope
- ❑ `>`
 - ❖ Hiển thị thuộc tính name của bean user trong context scope
- ❑ ``
 - ❖ Hiển thị tham số name

```

<ul>

```

```

    <li th:text="${message}" th:title="${message}"> </li>

```

```

    <li th:text="${session.message}"> </li>

```

```

    <li th:text="${application.message}"> </li>

```

```

</ul>

```

```

<ul>

```

```

    <li th:text="${bean.name}"/>

```

```

    <li th:text="${bean.salary}"/>

```

```

    <li th:text="${bean.gender}"/>

```

```

</ul>

```

```

<ul>

```

```

    <li th:text="${param.name}"/>

```

```

</ul>

```

```

<ul>
    <li title="Request variable">Request variable</li>
    <li>Session variable</li>
    <li>Context variable</li>
</ul>
<ul>
    <li>Nguyễn Chí Phèo</li>
    <li>50.0</li>
    <li>true</li>
</ul>
<ul>
    <li>Nguyễn Văn Tèo</li>
</ul>

```

```
<ul th:object="${book}">
```

```
<li th:text="*{title}">title</li>
```

```
<li th:text="${book.title}">title</li>
```

```
<li th:text="*{noOfPages}">number of pages</li>
```

```
<li th:text="*{author.name}">title</li>
```

```
<li th:text="*{publisher}">publisher</li>
```

```
<li th:text="*{pubYear}">published year</li>
```

```
</ul>
```

{...}: MESSAGE (I18N) EXPRESSIONS.

menu.properties

menu.home=Home

menu.about=About Us

menu.contact=Contact Us

menu.feedback=Feedback

menu.faq=FAQs

menu_vi.properties

menu.home=Trang chủ

menu.about=Giới thiệu

menu.contact=Liên hệ

menu.feedback=Góp ý

menu.faq=Hỏi - đáp

 Home

 About Us

 Contact Us

 Feedback

 FAQs

❑ Root relative URL (tương đối so với URL gốc - Webroot)

❖ <a th:href="@{/order/list}">...

➤ => ...

❑ Page Relative URL (tương đối so với URL hiện tại)

❖ <a th:href="@{../order/list}">...

➤ => ...

❖ <a th:href="@{order/list}">...

➤ => ...

❑ Protocol Relative and Absolute URL

❖ <a th:href="@{///www.poly.com/order/list}">...

➤ => ...

❖ <a th:href="@{https://www.poly.com/order/list}">...

➤ => ...

Parameters

```
<th:block th:with="x='X', y='Y'">
  <a th:href="@{/order/details(a=${x},b=${y})}">...</a>
  <a th:href="@{/order/details?a=${x}&b=${y}|">...</a>
  <a th:href="@{'/order/details?a=' + ${x} + '&b=' + ${y}|">...</a>
</th:block>
```

```
<a href="/order/details?a=X&b=Y">...</a>
```

Path Variables

```
<th:block th:with="x='X', y='Y'">
  <a th:href="@{/order/{a}/details/{b}(a=${x},b=${y})}">...</a>
  <a th:href="@{/order/${x}/details/${y}|">...</a>
  <a th:href="@{'/order/' + ${x} + '/details/' + ${y}|">...</a>
</th:block>
```

```
<a href="/order/X/details/Y">...</a>
```

❑ Fragment Expression được sử dụng để sao chép một file hoặc một fragment template được định nghĩa trước vào các vị trí mong muốn.

❖ `<div th:insert="~{/menu.html}">...</div>`

➤ Thay thế nội dung thẻ div bằng nội dung file **menu.html**

❖ `<div th:replace="~{/menu.html}">...</div>`

➤ Thay thế thẻ `<div>` bằng nội dung file **menu.html**

❖ `<div th:insert="~{/fragments.html :: menu}">...</div>`

➤ Thay thế nội dung thẻ div bằng fragment có tên là **menu** trong file **fragments.html**

❖ `<div th:replace="~{/fragments.html :: menu}">...</div>`

➤ Thay thế thẻ `<div>` bằng fragment có tên là **menu** trong **fragments.html**

DEMO





THYMELEAF OPERATORS

String

- +, |...\${}...|, [[expr]], [(expr)]

Arithmetic

- +, -, *, /, %

Logic

- and, or, not (!)

Comparison

- < (lt), <= (le), > (gt), >= (ge), == (eq), != (ne)

Conditional

- condition ? trueValue : falseValue


```
<ul th:with="x = 'X', y = '<b><i>Y</i></b>'">
```

```
<li th:utext="{x} + {y}"></li>
```

```
<li th:utext="|Value of y is {y}|"></li>
```

```
<li>Value of y is [{y}]</li>
```

```
<li>Value of y is [{y}]</li>
```

```
</ul>
```

- *XY*
- Value of y is *Y*
- Value of y is = <i>Y</i>
- Value of y is = *Y*

```
<ul>
```

```
<li>X<b><i>Y</i></b></li>
```

```
<li>Value of y is <b><i>Y</i></b></li>
```

```
<li>Value of y is = &lt;b&gt;&lt;i&gt;Y&lt;/i&gt;&lt;/b&gt;</li>
```

```
<li>Value of y is = <b><i>Y</i></b></li>
```

```
</ul>
```

```
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Student {
    String fullname = "Nguyễn Văn Tèo";
    Double marks = 9.5;
}
```

```
model.addAttribute("bean", new Student());
```

```
<ul th:object="${bean}">
    <li>Fullname: <b th:text="*{fullname}"></b></li>
    <li>Marks:
        <b th:text="*{marks}"></b>
        (<i th:text="*{marks >= 5.0 ? 'Passed' : 'Failed'}"></i>)
    </li>
    <li><b th:if="*{marks >= 9.0}">Golden Bee</b></li>
</ul>
```

- Fullname: **Nguyễn Văn Tèo**
- Marks: **9.5** (*Passed*)
- **Golden Bee**





DEMO



FLOW CONTROL ATTRIBUTES

❑ th:each

- ❖ th:each="item: \${iterable}"
- ❖ th:each="item, state: \${iterable}"
- ❖ th:each="entry: \${map}",
- ❖ th:each="entry, state: \${map}"

❑ th:if

- ❖ th:if="expr"
- ❖ th:unless="expr"

❑ th:switch

- ❖ <any th:switch="expr">
 - <any th:case="v1"/>
 - <any th:case="*/>
- ❖ </any>

```

<b th:if="*{marks >= 9.0}">Golden Bee</b>
<th:block th:unless="*{marks >= 9.0}">
  <b th:if="*{marks >= 8.5}">Excellent</b>
  <th:block th:unless="*{marks >= 8.0}">
    <b th:if="*{marks >= 7.5}">Good</b>
    <th:block th:unless="*{marks >= 7.5}">
      <th:block th:switch="*{marks >= 5.0}">
        <b th:case="true">Passed</b>
        <b th:case="*">Failed</b>
      </th:block>
    </th:block>
  </th:block>
</th:block>
</th:block>
</th:block>

```

	5		7.5		8.5		9	
Failed	Passed	Good	Excellent	Golden Bee				


```
<ul th:each="student: ${list}" th:object="${student}">
  <li>Fullname: <b th:text="*{fullname}"></b></li>
  <li>Marks: <b th:text="*{marks}"></b></li>
  <li>Grade:
    <b th:if="*{marks >= 9.0}">Golden Bee</b>
    <th:block th:unless="*{marks >= 9.0}">
      <b th:if="*{marks >= 8.5}">Excellent</b>
      <th:block th:unless="*{marks >= 8.0}">
        <b th:if="*{marks >= 7.5}">Good</b>
        <b th:unless="*{marks >= 7.5}">
          [[*{marks >= 5.0 ? 'Passed' : 'Failed'}]]
        </b>
      </th:block>
    </th:block>
  </li>
</ul>
```

- Fullname: **Tuấn**
- Marks: **9.5**
- Grade: **Golden Bee**
- Fullname: **Hạnh**
- Marks: **8.5**
- Grade: **Excellent**
- Fullname: **Phương**
- Marks: **7.5**
- Grade: **Good**
- Fullname: **Cường**
- Marks: **5.0**
- Grade: **Passed**
- Fullname: **Linh**
- Marks: **4.5**
- Grade: **Failed**



DEMO



THYMELEAF UTILITIES

- ❑ Thymeleaf định nghĩa sẵn các đối tượng servlet (***#request***, ***#response***, ***#session***, ***#servletContext***, ***#locale***), từ đó có thể gọi các phương thức để thực hiện công việc mong muốn.
 - ❖ `${#request.getRequestURI()}`
 - ❖ `${#session.getId()}`
 - ❖ `${#servletContext.getContextPath()}`
 - ❖ `${#locale.getLanguage()}`
- ❑ *Thymeleaf cho phép áp dụng quy ước JavaBean đối với các getter.*
 - ❖ `${#request.requestURI}`
 - ❖ `${#session.id}`
 - ❖ `${#servletContext.contextPath}`
 - ❖ `${#locale.language}`

❑ Thymeleaf cung cấp các đối tượng ngầm định chứa các phương thức tiện ích hỗ trợ xử lý trong Template

❖ **#messages**: xử lý tài nguyên đa ngôn ngữ

➤ Ex: `${#messages.msg('key', param1, param2)}`

❖ **#dates**: xử lý thời gian

➤ Ex: `${#dates.format(date, parttern)}`

❖ **#numbers**: Xử lý số

➤ Ex: `${#numbers.formatDecimal(marks, 0, 'WHITESPACE', 2, 'POINT')}`

❖ **#strings**: xử lý chuỗi

➤ Ex: `${#strings.capitalizeWords(str)}`

❖ **#arrays**: xử lý mảng

➤ Ex: `${#arrays.length(array)}`

❖ **#lists**: xử lý danh sách

➤ Ex: `${#lists.size(list)}`


❖ **#aggregates**: tổng hợp dữ liệu từ collection, arrays, map...

➤ Ex: `${#aggregates.sum(array)}`

```
<ul th:object="{student}">
  <li>Fullname:
    <b th:text="*{#strings.capitalizeWords(fullname)}"> </b> </li>
  <li>Marks:
    <b th:text="*{#numbers.formatDecimal(marks, 0, 'COMMA', 2, 'POINT')}"> </b> </li>
  <li>Birthday:
    <b th:text="*{#dates.format(dob, 'dd-MM-yyyy')}"> </b> </li>
  <li th:if="*{marks >= 9.0}">Grade: <b>Golden Bee</b> </li>
</ul>
```

@Data

```
public class Student {
    String fullname = "tuấn nguyên";
    Double marks = 9.5;
    Date dob = new Date();
}
```

- 
- Fullname: **Tuấn Nguyễn**
 - Marks: **9.50**
 - Birthday: **23-02-2021**
 - Grade: **Golden Bee**



DEMO

- ✓ Thymeleaf Template Structure and Syntax
 - ✓ Namespace, attribute, expressions
- ✓ Attributes Preceding
- ✓ Standard Expressions
 - ✓ `${}`, `*{}`, `#{}`, `@{}`, `~{}`
- ✓ Implicit Servlet Objects
 - ✓ `#session`, `#locale`
- ✓ Implicit Utility Objects
 - ✓ `#numbers`, `#dates`, `#strings`





FPT POLYTECHNIC

Thank you

