**BẢNG ĐÁNH GIÁ ĐIỂM THUYẾT TRÌNH**

**MÔN LẬP TRÌNH JAVA NÂNG CAO**

**LỚP CTK44-PM**

|  |  |
| --- | --- |
| **Thông tin đề tài thuyết trình** | **Thông tin sinh viên thực hiện đánh giá** |
| Tên đề tài: Bố cục trang Web bằng Thymeleaf Layout | MSSV của sinh viên đánh giá: 2012353 |
| Điểm tổng kết: Ngày trình bày:21/04/2023 | Họ tên sinh viên & ký xác nhận: Đoàn Cao Nhật Hạ |
| ***Thông tin sinh viên thuyết trình*** |
| MSSV: 2011433  Họ tên: Lê Văn Tài |

1. **ĐÁNH GIÁ THUYẾT TRÌNH**

**Thang điểm: Kém 0-5; Trung bình: 5-7; Khá 7-8; Tốt 8-9; Xuất sắc: 9-10**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Bảng chấm điểm nhóm thuyết trình** | | | | | | |
| Các chủ đề chính có được xác định? (A) | Slide thuyết trình có thiết kế tốt (B) | Giao tiếp với người nghe bằng ánh mắt (C) | Giao tiếp với người nghe bằng giọng nói (D) | Khả năng trình bày nội dung (E) | Thảo luận & Hỏi đáp (F) | Điểm trung bình = (A+B+C+D+E+F)/6 |
| **7** | **7** | **7** | **7** | **7** | **7** | **7** |

1. **BÀI TẬP NẮM KIẾN THỨC**

Dựa vào nội dung vừa trình bày, trả lời các câu hỏi sau:

1. Trình bày tóm tắt nội dung anh/ chị đã nắm được về chủ đề vừa trình bày?

**Thymeleaf**

Thymeleaf là một Java Template Engine. Có nhiệm vụ xử lý và generate ra các file HTML, XML, …

Các file HMTL do Thymeleaf tạo ra là nhờ kết hợp dữ liệu và template + quy tắc để sinh ra một file HTML chứa đầy đủ thông tin.

Việc của bạn là cung cấp dữ liệu và quy định template như nào, còn việc dùng các thông tin đó để render ra HTML sẽ do Thymeleaf giải quyết.

Cú pháp

Cú pháp của Thymeleaf sẽ là một attributes (Thuộc tính) của thẻ HTML và bắt đầu bằng chữ th: .

Với cách tiếp cận này, bạn sẽ chỉ cần sử dụng các thẻ HTML cơ bản đã biết mà không cần bổ sung thêm syntax hay thẻ mới như JSP truyền thống.

Ví dụ:

Để truyền dữ liệu từ biến name trong Java vào một thẻ H1 của HTML.

<h1 th:text="${name}"></h1>

Chúng ta viết thẻ H1 như bình thường, nhưng không chứa bất cứ text nào trong thẻ. Mà sử dụng cú pháp th:text="${name}" để Thymeleaf lấy thông tin từ biến name và đưa vào thẻ H1.

Thuộc tính th:text biến mất và giá trị biến name được đưa vào trong thẻ H1.

Đó là cách Thymeleaf hoạt động.

**Model & View Trong Spring Boot**

Model là đối tượng lưu giữ thông tin và được sử dụng bởi Template Engine để generate ra webpage. Có thể hiểu nó là Context của Thymeleaf

Model lưu giữ thông tin dưới dạng key-value.

Trong template thymeleaf, để lấy các thông tin trong Model. bạn sẽ sử dụng Thymeleaf Standard Expression.

1. ${...}: Giá trị của một biến.
2. \*{...}: Giá trị của một biến được chỉ định (sẽ giải thích ở dưới)

Ngoài ra, để lấy thông tin đặc biệt hơn:

1. #{...}: Lấy message
2. @{...}: Lấy đường dẫn URL dựa theo context của server

*${...} - Variables Expressions*

Trên Controller bạn đưa vào một số giá trị:

model.addAttribute("today", "Monday");

Để lấy giá trị của biến today, tôi sử dụng ${...}

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

Đoạn expression trên tương đương với:

ctx.getVariable("today");

*\*{...} - Variables Expressions on selections*

Dấu \* còn gọi là asterisk syntax. Chức năng của nó giống với ${...} là lấy giá trị của một biến.

Điểm khác biệt là nó sẽ lấy ra giá trị của một biến cho trước bởi th:object

Còn ${...} sẽ lấy ra giá trị cục bộ trong Context hay Model.

Vậy đoạn code ở trên tương đương với:

<div>

<p>Name: <span th:text="${session.user.firstName}"></span>.</p>

<p>Surname: <span th:text="${session.user.lastName}"></span>.</p>

</div>

*#{...} - Message Expression*

Ví dụ, trong file config .properties của tôi có một message chào người dùng bằng nhiều ngôn ngữ.

home.welcome=¡Bienvenido a nuestra tienda de comestibles!

Thì cách lấy nó ra nhanh nhất là:

<p th:utext="#{home.welcome}">Xin chào các bạn!</p>

Đoạn text tiếng việt bên trong thẻ p sẽ bị thay thế bởi thymeleaf khi render #{home.welcome}.

*@{...} - URL Expression*

@{...} xử lý và trả ra giá trị URL theo context của máy chủ cho chúng ta.

<!-- tương đương với 'http://localhost:8080/order/details?orderId=3' -->

<a href="details.html"

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

<!-- tương đương '/order/details?orderId=3' -->

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

<!-- tương dương '/gtvg/order/3/details' -->

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

Nếu bắt dầu bằng dấu / thì nó sẽ là Relative URL và sẽ tương ứng theo context của máy chủ của bạn.

1. Anh/ chị hãy đưa ra các nhận xét và góp ý cho sinh viên vừa trình bày?

Cần trình bày rõ nội dung hơn