

Thymeleaf – Presentation Layer Design

Thymeleaf

Thymeleaf merupakan Java Library berupa XML/XHTML/HTML5 template engine di server side baik untuk web maupun standalone environment. Thymeleaf digunakan untuk melakukan transformasi dari file XML/XHTML/HTML template menjadi sebuah file XML/XHTML/HTML yang utuh untuk ditampilkan. Tujuan atau fungsi utama dari Thymeleaf yaitu untuk memberikan fasilitas bagi developer untuk membuat template dengan elegant dan rapi sehingga developer dapat menggunakan template berulang kali secara dinamis, reusable.

Ciri menggunakan Thymeleaf adalah dengan menambahkan kode berikut ini pada file HTML.

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

Penggunaan For Looping

Pada thymeleaf dapat menggunakan iterasi untuk perulangan data. Seperti pada contoh berikut ini pada tampilan viewall student menggunakan perulangan th:each.

```
<html xmlns:th="http://www.thymeleaf.org">
  <head>
    <title>View All Students</title>
  </head>
  <body>
    <h1>All Students</h1>

    <div th:each="student, iterationStatus: ${students}">
      <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
      <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
      <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
      <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
      <a th:href="/student/delete/' + ${student.npm}">Delete Data</a><br/>
      <a th:href="/student/update/' + ${student.npm}">Update Data</a><br/>
      <hr/>
    </div>
  </body>
</html>
```

Eksresi iterationStatus berada pada atribut th:each berfungsi untuk melakukan iterasi dari kumpulan data pada suatu objek yang berada di salah satu atribut yang dimiliki suatu kelas Model. Dari contoh di atas, melakukan iterasi pada \${students}. Setelah atribut th:each, terdapat student yang merupakan property berisi data pada index ke-n sesuai dengan iterasi saat ini.

```
<div th:each="student, iterationStatus: ${students}" th:class="${iterationStatus.odd} ? 'odd'">
  <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
  <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
  <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
  <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
  <a th:href="/student/delete/' + ${student.npm}">Delete Data</a><br/>
  <a th:href="/student/update/' + ${student.npm}">Update Data</a><br/>
  <hr/>
</div>
```

Pertanyaan 1:

Apakah value yang dihasilkan dari \${iterationStatus.odd}? Tidak ada. Hasilnya sama saja.

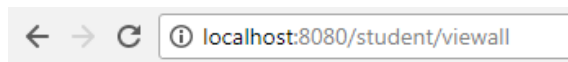
Penggunaan Conditional Expression pada Thymeleaf

Thymeleaf menyediakan conditional expression, salah satunya berupa `th:if=${...}` dan `th:unless=${...}`. Hal ini membantu dalam menampilkan data dengan kondisi tertentu seperti yang ditampilkan pada gambar berikut ini.

1. Tambahkan kode baris berikut ini pada file `viewall.html` untuk menambahkan kondisi dengan menggunakan `th:if` yaitu jika `gpa >= 3.49` maka akan menampilkan Cumlaude sedangkan jika tidak akan menampilkan Sangat memuaskan. Tag `th:if` adalah untuk memberikan kondisi dan `th:unless` akan memberikan kondisi selain `if`. Maka dari itu kondisinya harus sama dengan tag `th:if` seperti pada gambar berikut ini.

```
<div th:each="student, iterationStatus: ${students}" th:class="${iterationStatus.odd} ? 'odd'">
  <h3 th:text="'No. ' + ${iterationStatus.count}">No. 1</h3>
  <h3 th:text="'NPM = ' + ${student.npm}">Student NPM</h3>
  <h3 th:text="'Name = ' + ${student.name}">Student Name</h3>
  <h3 th:text="'GPA = ' + ${student.gpa}">Student GPA</h3>
  <h3 th:if="${student.gpa}>=3.49">Cum Laude!</h3>
  <h3 th:unless="${student.gpa}>=3.49">Sangat memuaskan!</h3>
  <a th:href="/student/delete/' + ${student.npm}">Delete Data</a><br/>
  <a th:href="/student/update/' + ${student.npm}">Update Data</a><br/>
  <hr/>
</div>
```

2. Coba run aplikasi Anda, lalu akses `localhost:8080/student/viewall`. Tampilan akan seperti gambar di bawah ini.



All Students

No. 1

NPM = 12345

Name = chaneK

GPA = 3.23

Sangat memuaskan!

[Delete Data](#)

[Update Data](#)

No. 2

NPM = 111111

Name = Lala

GPA = 3.67

Cum Laude!

[Delete Data](#)

[Update Data](#)

Pertanyaan 2:

Mengapa condition di dalam th:unless sama dengan di dalam th:if? Jelaskan jawaban Anda.

Jawab: Karena seperti yang telah dijelaskan di atas bahwa th:unless sudah merupakan negasi dari th:if sehingga isi kondisi keduanya harus sama.

Pertanyaan 3:

Coba ubah condition pada `th:unless=${student.gpa}>=3.49}` menjadi `th:unless=${student.gpa}<=3.48}`, lalu run kembali aplikasi Anda dan mengakses halaman yang sama. Apakah terjadi error? Jika iya, kenapa? Jelaskan alasan Anda. Jika tidak error, apakah hasilnya sesuai?

Jawab: Terjadi error seperti pada gambar di bawah ini

The value of attribute "th:unless" associated with an element type "h3" must not contain the '<' character.

Terjadi error karena atribut th:if dan th:unless tidak dapat berisi karakter < dan isi kondisi th:unless harus sama dengan th:if.

Pertanyaan 4:

Tuliskan cara lain Anda dengan penulisan conditional expression yang berbeda untuk mendapatkan hasil yang sama seperti gambar di atas. Hint: Gunakan ternary operator.

Jawab: Dengan menggunakan ternary operator dapat dilihat pada gambar berikut ini.

```
<h3 th:text="${student.gpa}>=3.49} ? 'Cum Laude': 'Sangat Memuaskan'"/></h3>
```

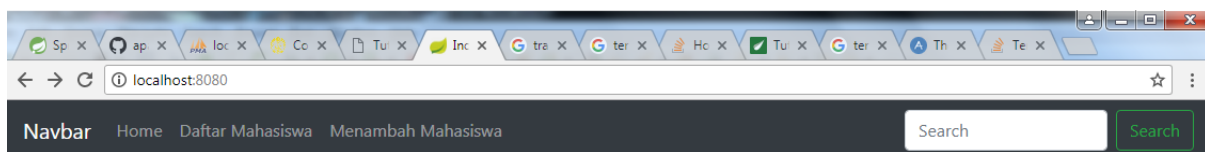
Static file

Spring Boot dan Thymeleaf memberikan fasilitas untuk menggunakan static file seperti file css, javascript, image, dan sebagainya. Static file tersebut disimpan dan diakses melalui folder static yang sudah tersedia secara default dari pembuatan project berbasis Thymeleaf. Sebelumnya download file bootstrap kemudian untuk tampilan index.html seperti pada gambar di bawah ini.

```
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Index</title>
<link rel="stylesheet" href="/bootstrap-4.0.0/dist/css/bootstrap.min.css" />
</head>
<body>
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
<a class="navbar-brand" href="#">Navbar</a>
<button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">
<ul class="navbar-nav mr-auto">
<li class="nav-item">
<a class="nav-link" href="/">Home</a>
</li>
<li class="nav-item">
<a class="nav-link" href="/student/viewall">Daftar Mahasiswa</a>
</li>
<li class="nav-item">
<a class="nav-link" href="/student/add">Menambah Mahasiswa</a>
</li>
</ul>
<form class="form-inline my-2 my-lg-0">
<input class="form-control mr-sm-2"
type="text"
placeholder="Search"
aria-label="Search" />
<button class="btn btn-outline-success my-2 my-sm-0" type="submit">
Search
</button>
</form>
</div>
</nav>
```

Akses halaman index dari aplikasi seperti gambar di bawah ini yaitu dengan tampilan NavBar dan menu – menu di bagian header.



Fragment

Fragment merupakan sebuah fitur dalam Thymeleaf yang memungkinkan developer untuk me-reuse source code dari halaman HTML-nya. Fitur ini memungkinkan Anda untuk memecah halaman HTML menjadi bagian-bagian kecil dan bagian tersebut dapat digunakan oleh halaman HTML lain. Code reuse ini berguna ketika developer menggunakan komponen-komponen yang sering dipakai di berbagai bagian dalam aplikasi yang dibuatnya. Contohnya adalah pembuatan header dan footer pada halaman htm tidak perlu dipanggil berulang kali, cukup dengan memanggil fragment yang akan digunakan maka akan menyesuaikan dengan halaman masing-masing.

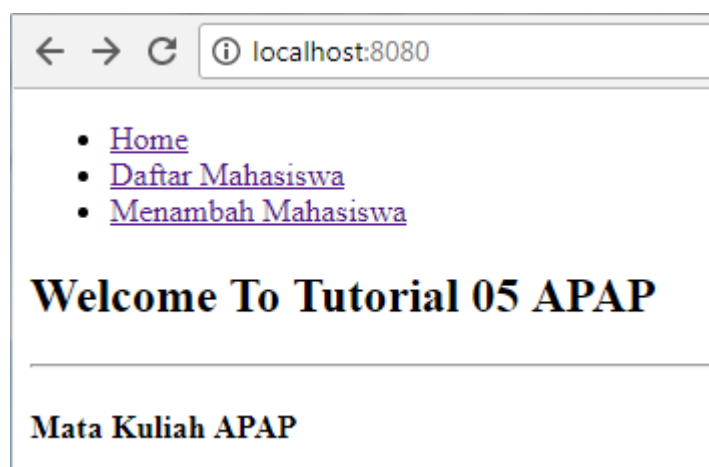
Buat sebuah halaman HTML bernama fragment.html yang berisi source code seperti dibawah ini dan disimpan pada folder fragments. Terdapat dua fragment yaitu header dan footer th:fragment = "header" dan "footer".

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Fragment</title>
</head>
<body>
<div th:fragment="header">
<ul>
<li><a href="/">Home</a></li>
<li><a href="/student/viewall">Daftar Mahasiswa</a></li>
<li><a href="/student/add">Menambah Mahasiswa</a></li>
</ul>
</div>
<div th:fragment="footer">
<hr/>
<h4>Mata Kuliah APAP</h4>
</div>
</body>
</html>
```

Kemudian pada halaman index.html perlu ditambahkan code seperti di bawah ini untuk memanggil fragment pada halaman index yaitu terdapat dua fragment yaitu header dan footer.

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Index</title>
</head>
<body>
<div th:replace="fragments/fragment :: header"></div>
<h2>Welcome To Tutorial 05 APAP</h2>
<div th:replace="fragments/fragment :: footer"></div>
</body>
</html>
```

Hasilnya



Pertanyaan 5:

Apa yang dimaksud dengan `th:replace="fragments/fragment :: header"` dan `th:replace="fragments/fragment :: footer"` pada file `index.html` yang Anda buat?

Jawab: Pemanggilan fragment pada halaman index yaitu header dan footer sehingga header dan footer `index.html` sesuai dengan yang telah didefinisikan pada fragment.

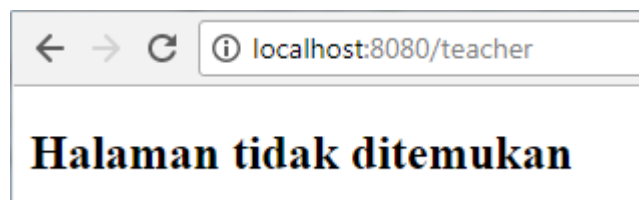
Error 404 Not Found Handler

Untuk mengatasi hal ketika halaman web yang dituju tidak ditemukan, kita dapat membuat handler melalui Thymeleaf. Pembuatan handler ini berguna agar pengguna mengetahui mengapa suatu error terjadi. Berikut ini adalah cara untuk membuat error 404 not found handler:

1. Buat sebuah folder dalam folder templates dengan nama error
2. Buat sebuah halaman HTML bernama `404.html` yang berisi source code seperti dibawah ini. File HTML tersebut dibuat pada folder error.

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
  <head>
    <title>404 Not found</title>
  </head>
  <body>
    <h2>Halaman tidak ditemukan</h2>
  </body>
</html>
```

Hasilnya



Pertanyaan 6:

Apa handler dengan metode ini juga dapat berlaku bagi error lain seperti error 500: internal server error?

Jawab: Ya. Metode handler ini dapat berlaku untuk error lain misalnya error 500.

Latihan

1. Penggunaan DataTables untuk mengubah tampilan pada menu daftar mahasiswa. DataTables adalah plugin jquery library, yang dapat membuat sebuah tabel lebih interaktif dan lebih baik (<https://datatables.net/>). Tahapan yang dilakukan adalah sebagai berikut.
 - Pertama download data table dari datatables.net/download seperti yang ditunjukkan pada gambar berikut.

Download

The best method for getting a hold of DataTables for use in your project depends upon your programming environment. The download builder below provides a simple method that you can use to build your own custom DataTables package - including only the software that you need, and providing options to have it hosted on the [DataTables CDN](#), download the package locally or install through a package manager such as *npm*, *yarn* or *bower*.

Compatibility

A [detailed compatibility table](#) shows the browser's that DataTables supports and also which features of the extensions can be used with the other extensions. A few features may not always be fully compatible with every other aspect of the software (due to overlapping functionality), so please take the time to review this table.

Step 1. Choose a styling framework

<input checked="" type="radio"/> DataTables	DataTables' default styling.	v1.10.16
<input type="radio"/> Bootstrap 3	One of the most popular styling libraries around.	v3.3.7

- Kemudian copy file.datatables dan jquery ke dalam folder js yang terdapat pada direktori bootstrap seperti pada gambar berikut ini

static > bootstrap-4.0.0 > dist > js

<input type="checkbox"/> Name	Date modified
bootstrap.bundle	3/17/2018 11:07 AM
bootstrap.bundle.js	3/17/2018 11:07 AM
bootstrap.bundle.min	3/17/2018 11:07 AM
bootstrap.bundle.min.js	3/17/2018 11:07 AM
bootstrap	3/17/2018 11:07 AM
bootstrap.js	3/17/2018 11:07 AM
bootstrap.min	3/17/2018 11:07 AM
bootstrap.min.js	3/17/2018 11:07 AM
.datatables	2/3/2018 4:21 PM
.datatables.min	2/3/2018 4:21 PM
jquery-3.2.1	9/13/2017 9:36 AM
jquery-3.2.1.min	9/13/2017 9:36 AM

- Untuk dapat menggunakan data tables maka perlu menambahkan pemanggilannya pada fragment.html agar dapat digunakan pada semua tampilan yang menggunakan data tables. Tambahkan sebuah fragment misalnya dengan nama fragment assets yang akan memanggil semua css dan javascript yang digunakan khususnya data tables.

```
<head th:fragment="assets">
  <title th:text="${title}"></title>
  <link rel="stylesheet" href="/bootstrap-4.0.0/dist/css/bootstrap.min.css" />
  <link rel="stylesheet" href="/bootstrap-4.0.0/dist/css/datatables.min.css" />
  <script type="text/javascript" src="/bootstrap-4.0.0/dist/js/jquery-3.2.1.min.js"></script>
  <script type="text/javascript" src="/bootstrap-4.0.0/dist/js/bootstrap.min.js"></script>
  <script type="text/javascript" src="/bootstrap-4.0.0/dist/js/datatables.min.js"></script>
  <script type="text/javascript">
    $(document).ready(function() {
      $('#studentTable').DataTable();

      $('.navbar-nav li').click(function(){
        $('.navbar-nav li').removeClass('active');
        $(this).addClass('active');
      });
    });
  </script>
</head>
```

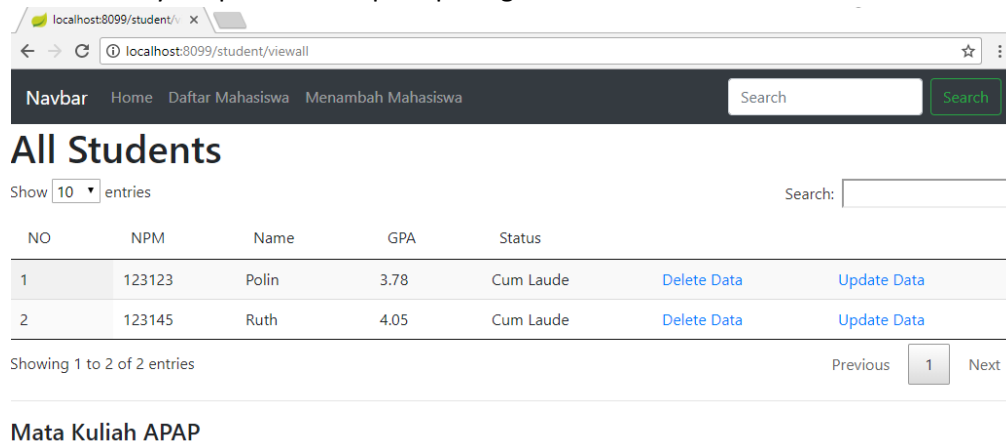
- Kemudian untuk dapat mengimplementasikan data tables maka perlu menambahkan table dan pemanggilan fragment assets pada halaman viewall seperti pada gambar berikut ini.

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head th:replace="fragments/fragment :: assets" />
<body>
  <div th:replace="fragments/fragment :: header"></div>
  <h1>All Students</h1>
  <table id="studentTable" class="display" style="width:100%">
    <thead>
      <tr>
        <td>NO</td>
        <td>NPM</td>
        <td>Name</td>
        <td>GPA</td>
        <td>Status</td>
        <td></td>
        <td></td>
      </tr>
    </thead>
    <tbody>
      <tr th:each="student, iterationStatus: ${students}">
        <td th:text="${iterationStatus.count}"></td>
        <td th:text="${student.npm}"></td>
        <td th:text="${student.name}"></td>
        <td th:text="${student.gpa}"></td>
        <td th:text="${student.gpa}>=3.49 ? 'Cum Laude': 'Sangat Memuaskan'"></td>
        <td><a th:href="/student/delete/' + ${student.npm}">Delete Data</a></td>
        <td><a th:href="/student/update/' + ${student.npm}">Update Data</a></td>
      </tr>
    </tbody>
  </table>
  <div th:replace="fragments/fragment :: footer"></div>
</body>
</html>

```

- Maka hasilnya dapat dilihat seperti pada gambar di bawah ini.



Untuk tampilan setiap halaman yang sama pada bagian header dan footer maka dapat menggunakan fragment header dan footer sehingga setiap halaman akan memiliki header dan footern yang sama.

2. Untuk dapat membuat tampilan header yang dinamis sesuai dengan halaman yang dibuka, maka pada halaman fragment.html perlu menambahkan kode `th:text = {title}` dengan nilai title yang akan didefinisikan pada controller seperti pada gambar berikut ini.

```

<div th:fragment="header">
  <title th:text="${title}"></title>

```

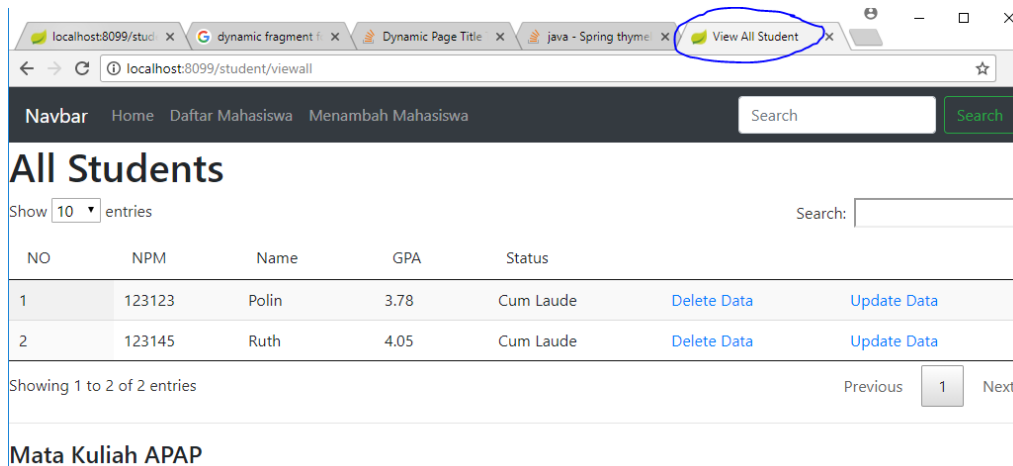
Kemudian pada controller setiap halaman tambahkan kode berikut untuk memberikan parameter title sesuai dengan halaman yang diinginkan dan harus didefinisikan di semua halaman html. Misalnya pada contoh dibawah ini pada method viewall yaitu dengan memberikan atribut title dan value view all student.


```

@RequestMapping("/student/viewall")
public String view (Model model)
{
    List<StudentModel> students = studentDAO.selectAllStudents ();
    model.addAttribute ("students", students);
    model.addAttribute("title", "View All Student");
    return "viewall";
}

```

Berikut ini hasil tampilan dengan title yang sama dengan yang telah didefinisikan pada controller.



The screenshot shows a web browser window with the address bar displaying 'localhost:8099/student/viewall'. The browser tab is titled 'View All Student'. The page features a navigation bar with links: Home, Daftar Mahasiswa, and Menambah Mahasiswa. Below the navigation bar, the main content area is titled 'All Students'. There is a search bar and a dropdown menu showing '10 entries'. A table displays student data with columns: NO, NPM, Name, GPA, and Status. The table contains two rows of data. At the bottom of the table, there are links for 'Delete Data' and 'Update Data' for each row. The footer of the page shows 'Showing 1 to 2 of 2 entries' and pagination controls with 'Previous', '1', and 'Next' buttons.

NO	NPM	Name	GPA	Status
1	123123	Polin	3.78	Cum Laude
2	123145	Ruth	4.05	Cum Laude

Mata Kuliah APAP

Hal yang sama perlu diimplementasikan pada seluruh file html yang ada agar semua tampilan (dinamis) memiliki title yang sesuai dengan yang didefinisikan pada masing-masing controller.