

WRITE UP

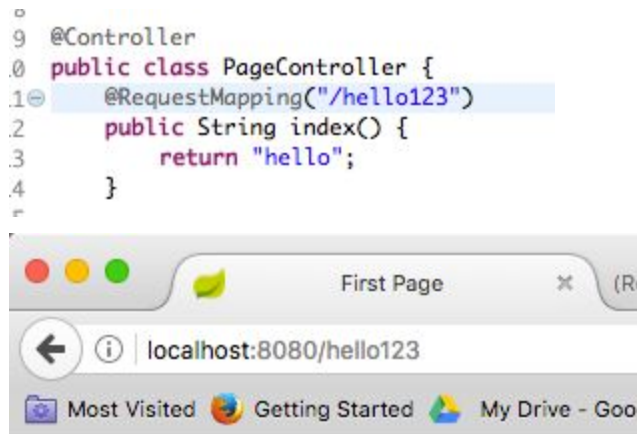
Tutorial 2 - Menggunakan Routing dan Controller dalam Project Spring Boot

Pada tutorial kali ini, kita diajarkan menggunakan Java untuk melakukan programming berbasis web. Framework yang digunakan adalah Spring Framework sebagai salah satu framework berbasis Java yang populer.

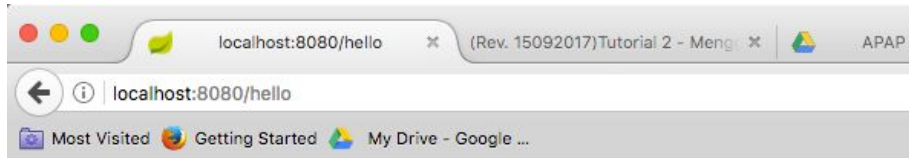
Latihan Project Hello World

1. Ganti baris tersebut menjadi `@RequestMapping("/hello123")`

Tidak terjadi compile error, web tetap bisa dibuka seperti normal di `/hello123`, namun kalo kita buka kembali `/hello`, akan error not found.



Hello, world!



Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Fri Sep 15 23:49:05 WIB 2017

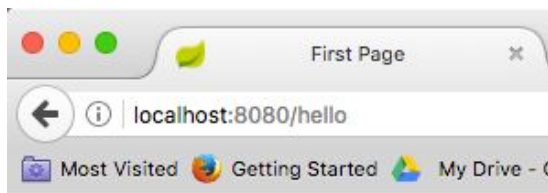
There was an unexpected error (type=Not Found, status=404).

No message available

2. Ganti nama method index() dengan nama method hello()

Tidak terjadi compile error, web juga normal normal saja. Hal ini karena apapun nama method dibawah anotasi request param akan tetap dijalankan (akan saya jelaskan detilnya dibawah).

```
public String hello ()  
{  
    return "hello";  
}
```

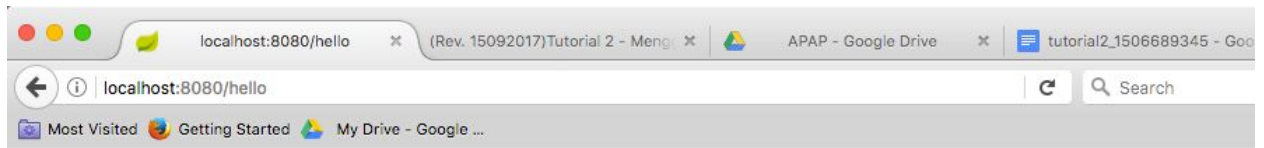


Hello, world!

3. Ganti string return type menjadi return "hello123";

Tidak compile error, namun saat buka /hello malah jadi error 500 Internal server error. Ini dikarenakan sistem akan mencari template hello123.html yang ternyata tidak ditemukan.

```
@Controller  
public class PageController {  
    @RequestMapping("/hello")  
    public String index() {  
        return "hello123";  
    }  
}
```



Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Fri Sep 15 23:52:43 WIB 2017

There was an unexpected error (type=Internal Server Error, status=500).

Error resolving template "hello123", template might not exist or might not be accessible by any of the configured Template Resolvers

Latihan Request Parameter

1. Ubah nilai anotasi RequestMapping dari `"/greeting"` menjadi `"/hello/greeting"`, Buka `localhost:8080/hello/greeting?name=chanek`

Baik-baik saja, tidak compile error, dan web lancar. Ini karena walaupun kita sudah ganti routesnya di request mapping, saat kita buka url, urlnya juga kita ganti menjadi `/hello/greeting`.

2. Akses `localhost: 8080/greeting`, Ubah header method greeting menjadi seperti berikut

```
public String greeting (@RequestParam(value = "name", required = false) String name, Model model)
```

Stop Spring Boot yang sedang berjalan, run kembali, buka `localhost: 8080/greeting`.

Awal mula saya buka `/greeting`, muncul error sebab request param name tidak dikirimkan.

There was an unexpected error (type=Bad Request, status=400).

Required String parameter 'name' is not present

Setelah diberikan `required false` hasilnya tidak compile error, webnya juga berjalan dengan baik, padahal ketika kita buka `/greeting`, kita tidak mengirimkan request param apapun. Hal ini dikarenakan kita menambah `required = false` yang membuat request param kita menjadi optional untuk dikirim, artinya kita bisa saja membuka `/greeting` tanpa mengirim param apapun. Sebagai gantinya, variable name akan bernilai null karena tidak diberi default value dan juga tidak di handle jika nilainya null.

Selamat datang null!

3. Ubah header method greeting menjadi seperti berikut

```
public String greeting (@RequestParam(value = "name", required = false,
defaultValue = "dunia") String name, Model model)
```

Stop Spring Boot yang sedang berjalan, run kembali, dan buka localhost: 8080/greeting. Karena sebelumnya tidak dihandle default value, maka sekarang dihandle. Hasilnya adalah seperti screenshot dibawah. Tidak lagi null sebab jika request param tidak dikirim, akan diberi nilai default "dunia".

Selamat datang dunia!

4. Perhatikan bahwa pada berkas greeting.html, tag paragraf yang kita tambahkan adalah sebagai berikut:

```
<p th:text="'Selamat datang ' + ${name} + '!'">Sapaan untuk user</p>
```

Mengapa sapaan untuk user tidak pernah ditampilkan

Karena isi dari th:text melakukan override terhadap isi dari tag p itu sendiri. Hal ini bisa jadi bawaan dari framework spring yang kita gunakan.

Latihan Path Variable

1. Akses localhost:8080/greeting/.

Tidak compile error, namun pas buka webnya, muncul error sebab ketika gunakan PathVariable, kita hanya buka /greeting (bukan /greeting/{name}), tidak ada parameter yang dikirimkan sehingga menjadi error.

Ubah method greetingPath menjadi seperti berikut

```
@RequestMapping(value = {"/greeting", "greeting/{name}"})
public String greetingPath(@PathVariable Optional<String> name, Model model) {
    if (name.isPresent()) {
        model.addAttribute("name", name.get());
    } else {
        model.addAttribute("name", "dengklek");
    }
    return "greeting";
}
```

Akses localhost:8080/greeting/ Pertanyaan: Apa hasilnya?

Menjadi tidak error lagi webnya, hal ini karena sudah diberi tanda Optional yang artinya name bisa saja tidak dikirimkan. Pada method diberi if jika name tidak dikirim, maka yang dikirim ke templates namanya adalah dengklek, hal ini membuat ketika web dibuka muncul seperti...

Selamat datang dengklek!

Akses localhost:8080/greeting/chanek Pertanyaan: Apa hasilnya?

Karena pada kasus ini name dikirimkan, sesuai ifnya, name yang dikirim ke html akan menjadi name yang dibuka di url browser. Tampilan hasilnya seperti....

Selamat datang chanek!

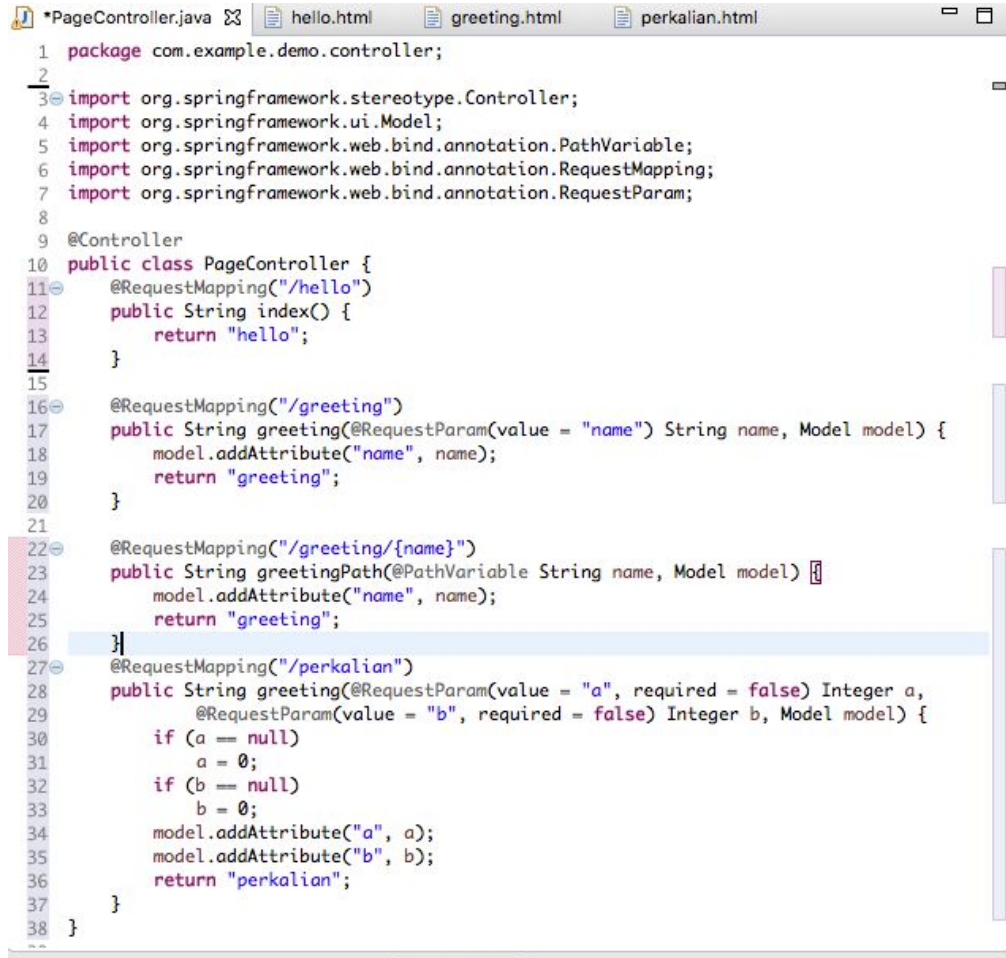
Ps. Semua hasil latihan saya kembalikan lagi ke awal (sebelum dijalankannya eksperimen2 diatas).

What I've learned

Salah satu komponen di framework ini adalah controller, tempat semua logic dari program kita berada. Kita bisa menggunakan RequestMapping untuk mendefinisikan alamat url dari web kita. Misal kita buat **@RequestMapping("/hello")** maka ketika kita run dan buka web kita di **/hello**, akan dijalankan logic pada method yang terdapat dibawah anotasi request mapping tersebut (tidak peduli apapun nama methodnya).

Method tersebut memiliki return type berupa String. String yang direturn merupakan nama dari templates html yang sudah kita buat di folder templates. Misal pada method index() yang mendefinisikan RequestMapping /hello, returnnya "hello", maka akan ditampilkan templates hello.html (jika ditemukan). Jika templatesnya tidak ditemukan, maka akan menjadi error.

Berikut adalah code PageController yang saya buat, hasil akhir setelah eksperimen2 diatas sudah saya kembalikan lagi ke awal.



```

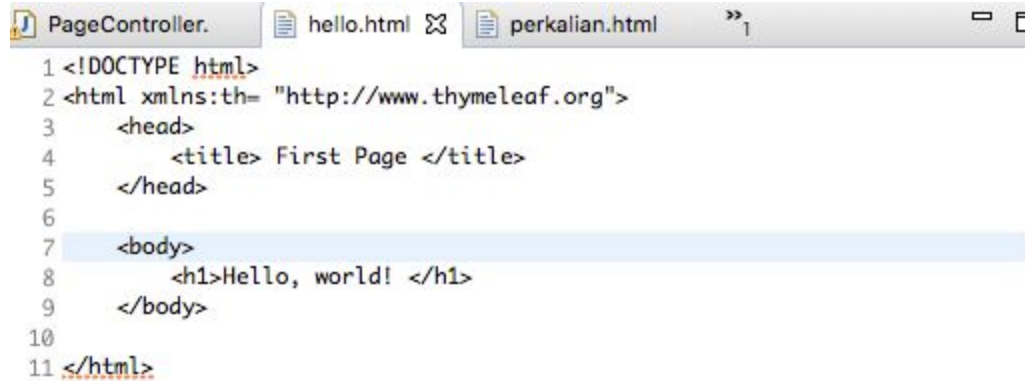
1 package com.example.demo.controller;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.ui.Model;
5 import org.springframework.web.bind.annotation.PathVariable;
6 import org.springframework.web.bind.annotation.RequestMapping;
7 import org.springframework.web.bind.annotation.RequestParam;
8
9 @Controller
10 public class PageController {
11     @RequestMapping("/hello")
12     public String index() {
13         return "hello";
14     }
15
16     @RequestMapping("/greeting")
17     public String greeting(@RequestParam(value = "name") String name, Model model) {
18         model.addAttribute("name", name);
19         return "greeting";
20     }
21
22     @RequestMapping("/greeting/{name}")
23     public String greetingPath(@PathVariable String name, Model model) {
24         model.addAttribute("name", name);
25         return "greeting";
26     }
27
28     @RequestMapping("/perkalian")
29     public String greeting(@RequestParam(value = "a", required = false) Integer a,
30         @RequestParam(value = "b", required = false) Integer b, Model model) {
31         if (a == null)
32             a = 0;
33         if (b == null)
34             b = 0;
35         model.addAttribute("a", a);
36         model.addAttribute("b", b);
37         return "perkalian";
38     }
39 }

```

Di method tersebut, kita juga bisa menerima parameter baik itu yang berupa **RequestParam** (seperti GET Request) dengan contoh **/namaRoute?param1=value¶m2=value**, ataupun **PathVariable** dengan contoh **/namaRoute/param1/param2**. Cara menerima parameter tersebut seperti kodingan diatas. Setelah parameter kita terima, terkadang kita ingin melakukan passing data ke template html kita. Passing data dapat dilakuka menggunakan object Model dengan cara `model.addAttribute("namaDataYangDiPass", valueData)`. Setelah hal tersebut dilakukan, maka variable yang dipass dapat diakses di html kita.

Kita juga bisa membuat suatu param menjadi optional dengan menambahkan `required = false`, dan membuat struktur data yg diterima non-primitive type agar dapat menerima nilai null (seandainya param tidak dikirimkan di browser).

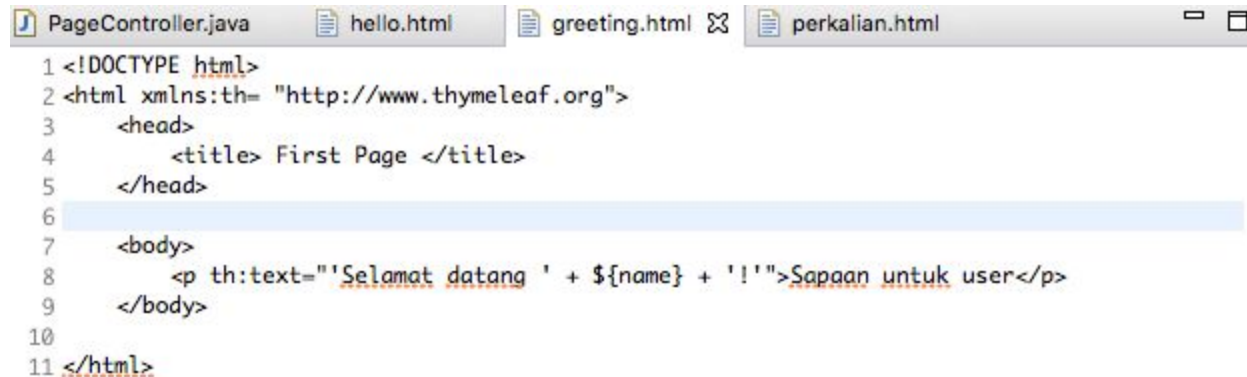
Contoh dari isi hello.html, salah satu template yang simple tanpa menerima variable apapun dari controller.



```
PageController. hello.html perkalian.html »_1
1 <!DOCTYPE html>
2 <html xmlns:th= "http://www.thymeleaf.org">
3   <head>
4     <title> First Page </title>
5   </head>
6
7   <body>
8     <h1>Hello, world! </h1>
9   </body>
10
11 </html>
```

Greeting.html

Dapat kita lihat pada html dibawah ini, menerima variable bernama name. Variable diterima dari controller (baik itu yang dari RequestParam maupun yang PathVariable). Variable ditampilkan menggunakan format seperti dibawah ini.



```
PageController.java hello.html greeting.html perkalian.html
1 <!DOCTYPE html>
2 <html xmlns:th= "http://www.thymeleaf.org">
3   <head>
4     <title> First Page </title>
5   </head>
6
7   <body>
8     <p th:text="'Selamat datang ' + ${name} + '!'">Sapaan untuk user</p>
9   </body>
10
11 </html>
```

Perhatikan bahwa kata2 sapaan untuk user disitu tidak ditampilkan, yang ditampilkan hanya yang terdapat dalam th:text saja.

Perkalian.html

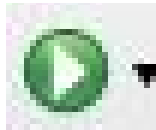
Ini merupakan item utama dari latihan pada tutorial ini. Html ini akan menerima 2 variable yang dikirim dari controller, yaitu a dan b. Variable tersebut, selain bisa ditampilkan, juga bisa dilakukan operasi matematika seperti perkalian.



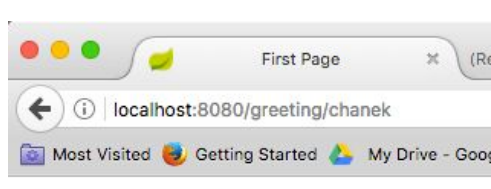
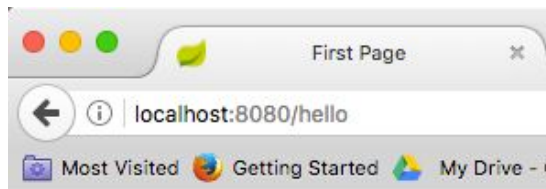
```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3   <head>
4     <title> First Page </title>
5   </head>
6
7
8   <body>
9     <h1>Perkalian </h1>
10    <p th:text='${a} + \'X\' + ${b} + \'=\' + ${a} * ${b}\'> Hasil perkali
11  </p>
12
13 </body>
14 </html>
```

Tombol run

Setelah semua code kita rampung, kita bisa menjalankan project kita di local dengan cara run project seperti yang dijelaskan di soal tutorial, atau dengan klik tombol run.

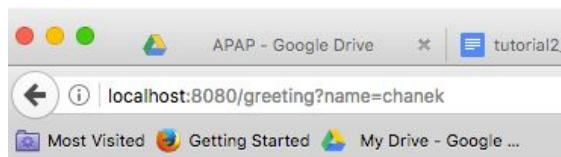


Setelah selesai proses compile dan run (bisa dilihat di log eclipse), kita bisa akses pagennya di localhost:8080. Sebagai contoh beberapa point yang saya coba di local saya.



Selamat datang chanek!

Hello, world!

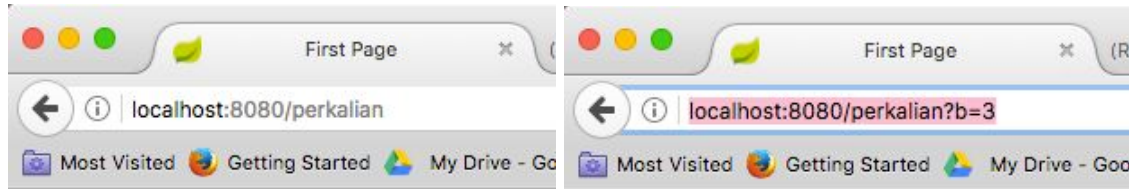


Selamat datang chanek!

Latihan Perkalian

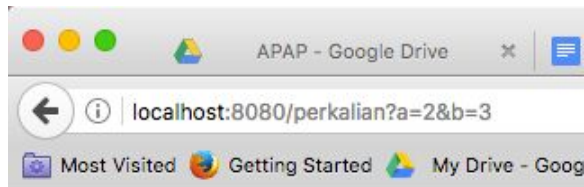
1. Buatlah satu halaman baru untuk melakukan operasi perkalian.
2. Halaman ini menerima 2 buah parameter dari query string yaitu a dan b.
3. Pada view tampilkan hasil perkalian a dan b.
4. Jika a atau b tidak ada, nilai defaultnya 0.
5. Contoh hasil tampilannya

Screenshot browser perkalian



Perkalian

0X0=0



Perkalian

0X3=0

Perkalian

2X3=6

Screenshot code controller

```
1  
@RequestMapping("/perkalian")  
public String greeting(@RequestParam(value = "a", required = false) Integer a,  
    @RequestParam(value = "b", required = false) Integer b, Model model) {  
    if (a == null)  
        a = 0;  
    if (b == null)  
        b = 0;  
    model.addAttribute("a", a);  
    model.addAttribute("b", b);  
    return "perkalian";  
}
```

Screenshot code html



The screenshot shows a web browser window with three tabs: 'PageController.', 'hello.html', and 'perkalian.html'. The 'perkalian.html' tab is active, displaying a Thymeleaf template. The code is as follows:

```
1 <!DOCTYPE html>
2 <html xmlns:th="http://www.thymeleaf.org">
3   <head>
4     <title> First Page </title>
5   </head>
6
7
8   <body>
9     <h1>Perkalian </h1>
10    <p th:text="${a} + 'X' + ${b} + '=' + ${a} * ${b}"> Hasil perkali
11  </p>
12
13 </body>
14 </html>
```

Perhatikan bahwa jika parameter tidak dikirimkan, default valuenya menjadi 0 (dengan cara yang sudah saya jelaskan di atas). Dengan ini, sudah sesuai dengan requirement yang dibutuhkan oleh soal Latihan Tutorial 2.

Pada tutorial kali ini saya menjadi belajar tentang framework berbasis java spring ini, dan bisa mengaplikasikan beberapa komponen seperti controller, templates, dan operasi-operasi sederhana untuk membuat apps sederhana.

=====

=