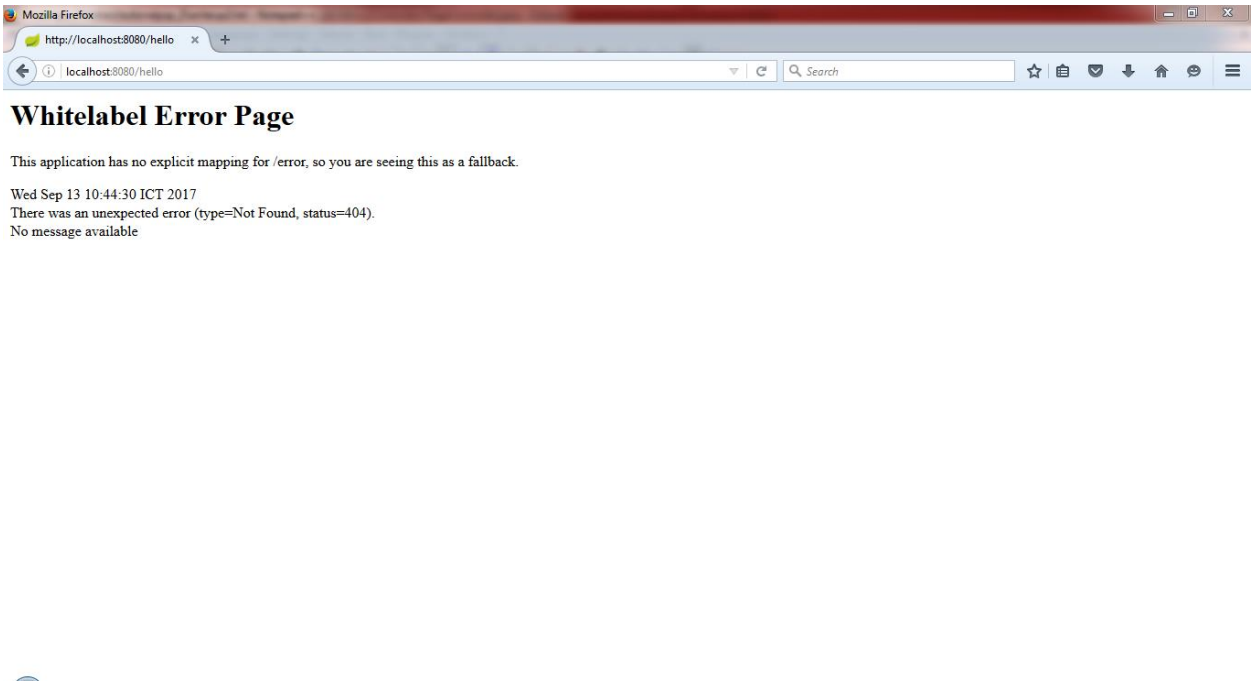
Iman Alfathan Yudhanto
1406623524
APAP-A

Write-up dan Latihan Tutorial 2

Latihan

1. Tidak terjadi compile error. Setelah dirun ulang, terjadi white label error. Penyebabnya adalah salah akses url. Url yang diakses harus sesuai dengan yang ditulis di controller.



2. Tidak terjadi compile error. Page tersebut masih bisa muncul. Untuk kasus ini, nama method tidak mempengaruhi pemanggilan url tersebut.

Hello, world!

3. Tidak terjadi compile error. Setelah dirun ulang, terjadi white label error resolving template "hello123", template might not exist or might not be accessible by any of the configured Template Resolvers. String tersebut menandakan template yang akan dipanggil.

Whitelabel Error Page

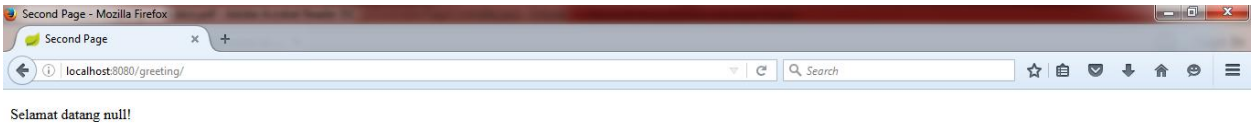
This application has no explicit mapping for /error, so you are seeing this as a fallback.

Wed Sep 13 10:49:16 ICT 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

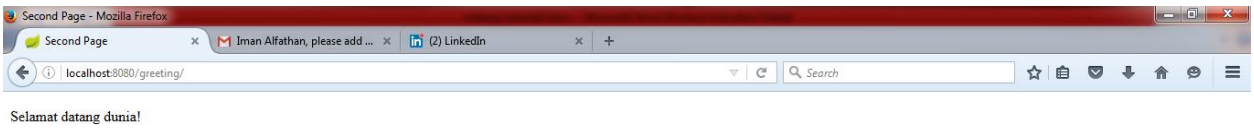
Latihan Request Parameter

1. Dapat mengembalikan hasil sesuai parameter yang dipanggil. Dalam kasus ini, jika variable name kosong, maka akan muncul error karena pada controller dibutuhkan sebuah parameter.

2. Tidak terkena white label error dan kode dapat dijalankan karena ada bagian `required = false` yang membolehkan bagian name kosong.



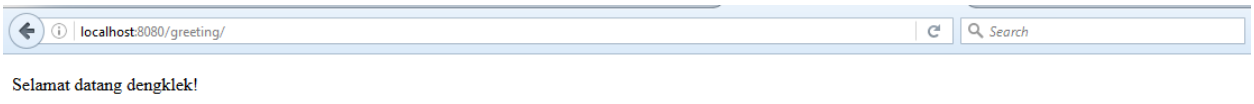
3. Tidak terkena white label error dan kode dapat dijalankan. Ketika bagian nama di url kosong, maka secara default *variable* nama menjadi 'dunia'.



4. Penyebabnya adalah karena tulisan tersebut tertimpa oleh th:text.

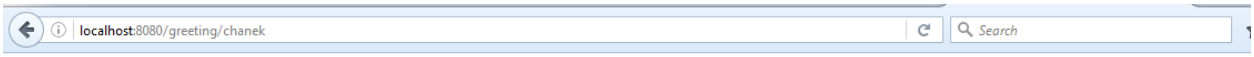
Latihan Path Variabel

1. Akses **localhost:8080/greeting/**
Pertanyaan: Apa hasilnya?



Dapat menampilkan hasil walau parameter name tidak ditulis.

Akses **localhost:8080/greeting/chanek**
Pertanyaan: Apa hasilnya?
Dapat menampilkan hasil dengan parameter yang ditulis. Parameter name dapat ditampilkan.



Selamat datang chanek!

Write-up Tutorial

Pada tutorial ini diajari membuat dan menggunakan Controller pada Spring Boot dan Pengenalan dasar routing pada Spring Boot. Jadi cara kerja controller pada tutorial ini adalah sebagai pemanggil dan penghubung via url.

```
@RequestMapping("/hello")
	public String index() {
		return "hello";// dia nganggep template sebagai kumpulan string. Jadi
					// dia langsung manggil semua stringnya.
	}
```

Pada controller tersebut @RequestMapping berfungsi untuk membuat URL yang akan diakses via browser. Setiap method yang ada di Controller dapat memanggil html yang diakses atau objek berupa *response entity.* Pada method di atas, fungsi index memanggil html yang bernama "hello". Jadi method tersebut menganggap html/*template* sebagai kumpulan string. Jadi dia langsung memanggil stringnya (berhubung fungsi tersebut me-*return* string). Jika fungsi tersebut me-*return* html yang tidak ada di template, maka akan terjadi white label error.
Penulisan URL pada @RequestMapping dapat berupa `@RequestMapping("/greeting")`
Atau `@RequestMapping ( "/greeting/{name}")`

```
@RequestMapping("/greeting")
	public String greeting(@RequestParam(value = "name", required = false,
defaultValue = "dunia") String name,
			Model model) {
		model.addAttribute("name", name);
		return "greeting";// dia nganggep template sebagai kumpulan string. Jadi
					// dia langsung manggil semua stringnya.
	}
```

Pada fungsi kedua di atas, terdapat @requestParam(value = "name", required = false, defaultValue = "dunia"). @RequestParam berfungsi untuk memasukkan input ke *controller.* Value menyatakan *variable* untuk menginput. Required merupakan Boolean untuk menentukan apakah parameter harus diinput atau tidak. Jika false, maka parameter tidak perlu diinput. Sedangkan defaultValue menentukan input *default* jika parameter bernilai kosong/tidak dimasukkan. Model.addAttribute untuk memasukkan variable yang diinput ke dalam html. Di method pada fungsi di atas, method tersebut memasukan *variable* name yang telah diinput ke dalam html. Jadi variable ${name} yang ada di dalam html akan ter-*replace* oleh *variable* yang telah diinput. Jadi contoh penulisan url-nya menjadi **localhost:8080/greeting?name=yudha**.

```
@RequestMapping ( "/greeting/{name}")
```

Sedangkan untuk Requet Mapping seperti itu, berarti nama variable yang ingin diinput langsung bisa ditempatkan langsung di url. Jika url seperti contoh di atas, maka membutuhkan @PathVariable. Dalam Spring Boot, {name} merupakan path variable yang dapat diolah oleh controller. Jadi contoh penulisan url-nya menjadi **localhost:8080/greeting/yudha**.