

## Part 1: Setup

### 1. Instalasi Gradle

```
C:\Users\asus>gradle -v

Welcome to Gradle 4.7!

Here are the highlights of this release:
- Incremental annotation processing
- JDK 10 support
- Grouped non-interactive console logs
- Failed tests are re-run first for quicker feedback

For more details see https://docs.gradle.org/4.7/release-notes.html

WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.codehaus.groovy.reflection.CachedClass (file:/C:/Gradle/gradle-
ize())
WARNING: Please consider reporting this to the maintainers of org.codehaus.groovy.reflection.CachedClass
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release

-----
Gradle 4.7
-----

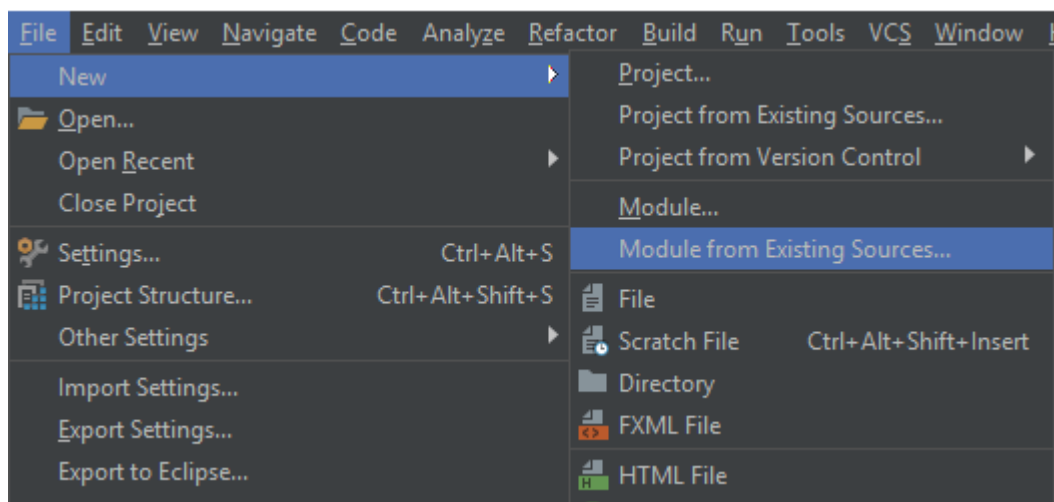
Build time:   2018-04-18 09:09:12 UTC
Revision:    b9a962bf70638332300e7f810689cb2febbd4a6c

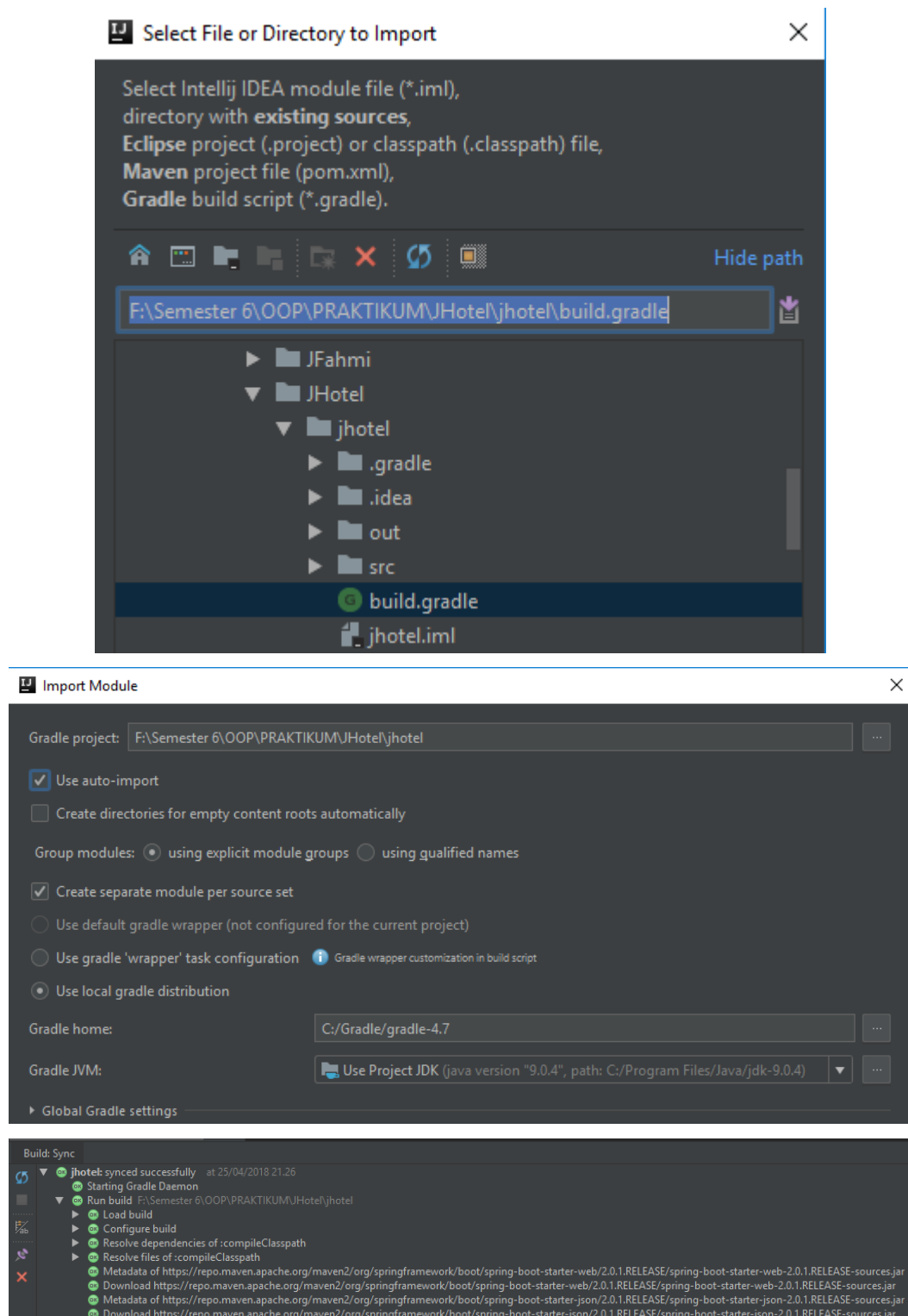
Groovy:      2.4.12
Ant:         Apache Ant(TM) version 1.9.9 compiled on February 2 2017
JVM:        9.0.4 (Oracle Corporation 9.0.4+11)
OS:         Windows 10 10.0 amd64

C:\Users\asus>
```

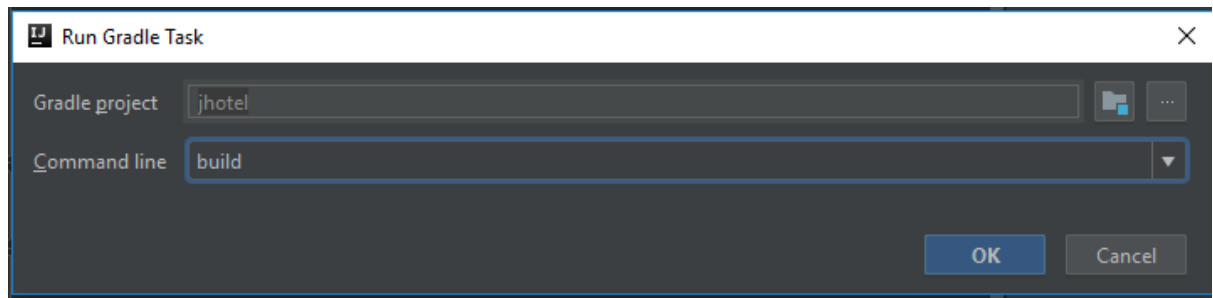
### 2. Backup Folder → Sudah

### 3. Instalasi Gradle Pada IntelliJ untuk project JHotel

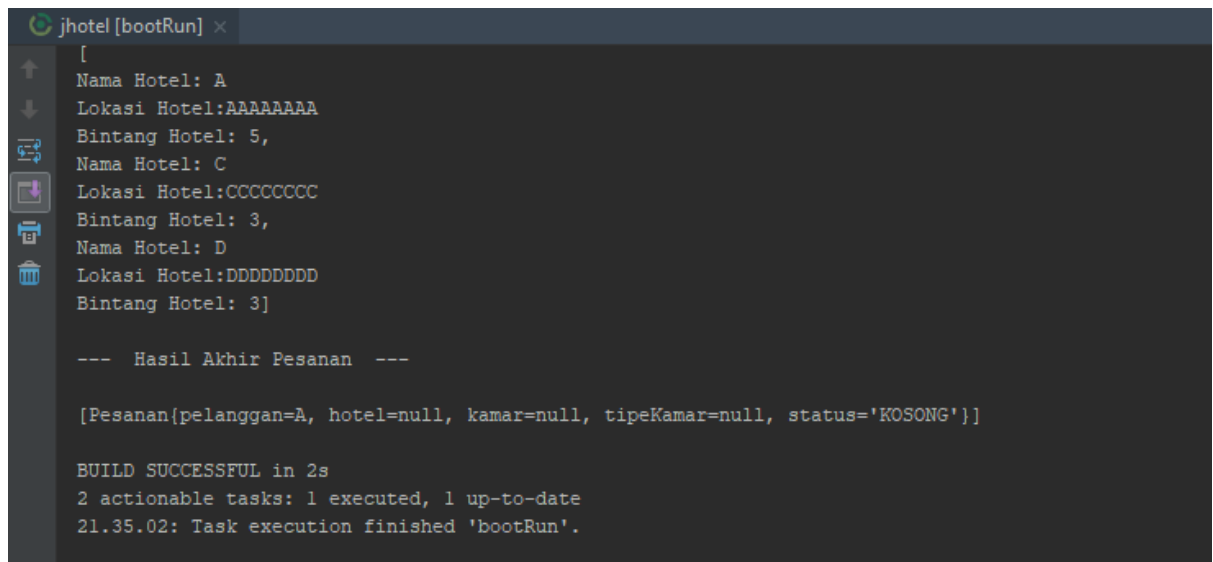




#### 4. Pada IntelliJ buka dan konfigurasi gradle



## 5. Run Gradle



## 6. Membuat Folder Controller

: PC > Kuliah (F:) > Semester 6 > OOP > PRAKTIKUM > JHotel > jhotel > src > main > java > jhotel				
Name	Date modified	Type	Size	
controller	25/04/2018 21.36	File folder		
Administrasi	18/04/2018 12.24	JAVA File	3 KB	
Customer	19/04/2018 18.08	JAVA File	4 KB	

## Part 2: Coding

1. Menambahkan package pada semua class
  - Class selain CustomerController.java

TipeKamar.java × JHotel.java × DatabaseHotel.java × RoomSudahAdaException.java ×


```
package jhotel;  
import java.lang.Exception;
```

- Class CustomerController.java

```
package jhotel;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
```

Sudah.

2. Edit class JHotel.java, Tambahkan CustomerController.java dari github ke folder controller yang telah dibuat sebelumnya, lalu bootRun program.



```
jhotel [bootRun] ×
21.48.54: Executing task 'bootRun'...

> Task :compileJava
> Task :processResources NO-SOURCE
> Task :classes

> Task :bootRun

      .
     /\  / ____' _ _ _ _ _ ( ) _ _ _ _ _ \ \ \ \ \
    ( ) \___| ' _ | ' _ | | ' _ \ \ \ \ \ \ \ \ \ \
   \ \ / ____| | _ | | | | | | | | ( | | ) ) ) )
    ' | ____| . _ | | | | | | \_\, | / / / / /
   =====|_|=====|___/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_
:: Spring Boot ::                (v2.0.1.RELEASE)
```

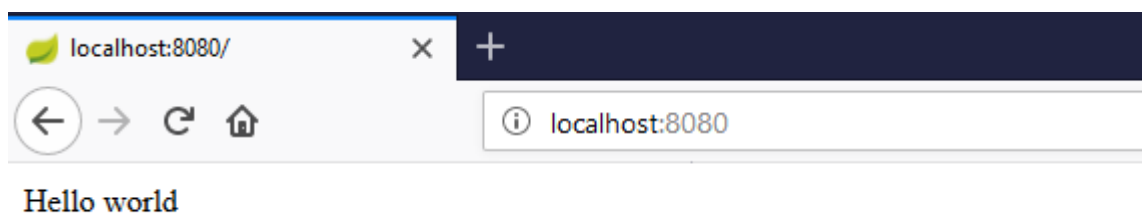
Jika berhasil muncul log seperti gambar dibawah

```

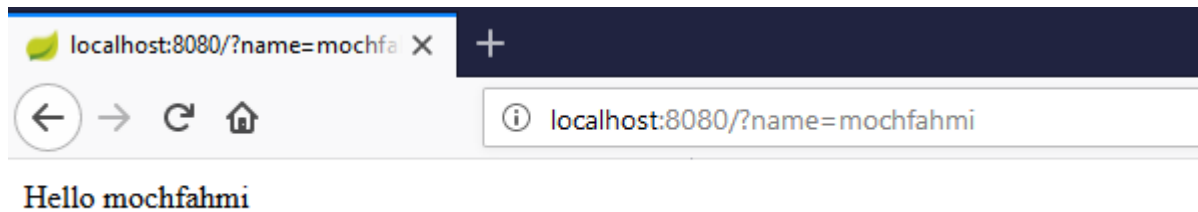
2018-04-25 23:47:28.250 INFO 7956 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2018-04-25 23:47:28.250 INFO 7956 --- [main] javaxel.JHotel : Started JHotel in 3.834 seconds (JVM running for 4.339s)
2018-04-25 23:47:39.463 INFO 7956 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring FrameworkServlet 'dispatcherServlet'
2018-04-25 23:47:39.463 INFO 7956 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : FrameworkServlet 'dispatcherServlet': initialization started
2018-04-25 23:47:39.487 INFO 7956 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : FrameworkServlet 'dispatcherServlet': initialization completed in 24 ms

```

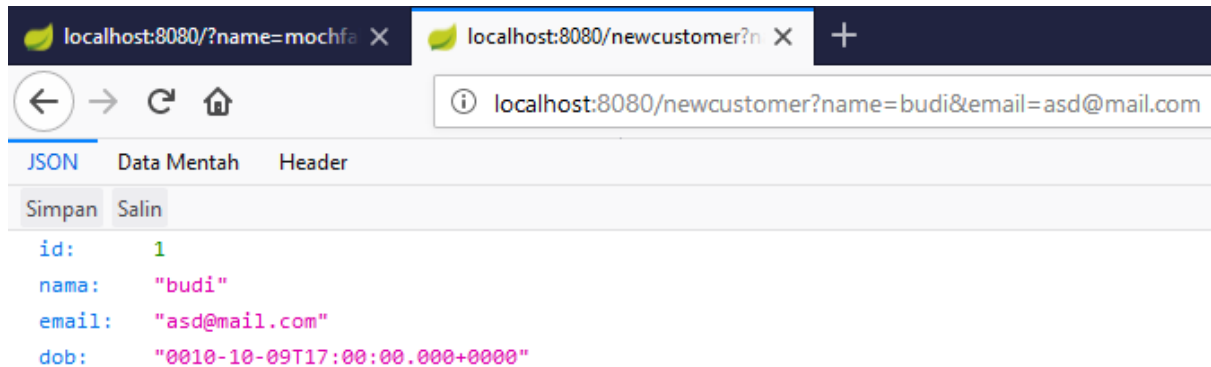
Bukalah `http://localhost:8080` pada browser



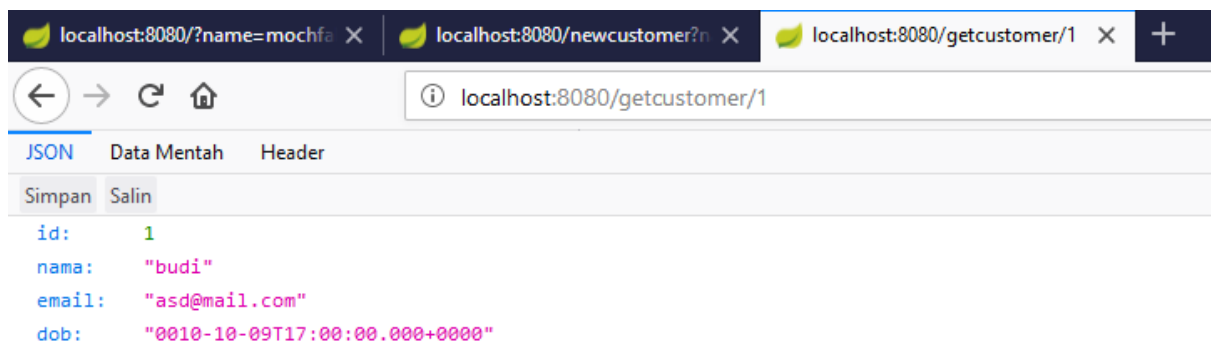
Bukalah URL `http://localhost:8080/?name=<nama_anda>`



3. Masukkan URL <http://localhost:8080/newcustomer?name=budi&email=asd@mail.com>



4. Masukkan URL `http://localhost:8080/getcustomer/{id_anda}` ke browser, dengan `{id_anda}`



5. Ubah method newcust() beserta annotation @RequestMapping menjadi POST. Buka URL `http://localhost:8080/newcustomer` menggunakan Postman dan masukkan data

```
@RequestMapping(value = "/newcustomer", method = RequestMethod.POST) public Customer newCust(@RequestParam Customer customer) {
    Customer customer = new Customer(name, tanggal, bulan, tahun, email);
    try {
        DatabaseCustomer.addCustomer(customer);
    } catch (Exception ex) {
        ex.getMessage();
        return null;
    }
}
```

POST ▼ http://localhost:8080/newcustomer

Authorization ● Headers (3) **Body** ● Pre-request Script Tests

☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary

Key	Value
<input checked="" type="checkbox"/> name	coba
<input checked="" type="checkbox"/> email	coba@coba.com
New key	Value

Body Cookies Headers (3) Test Results

Pretty Raw Preview JSON ▼

```

1 {
2   "id": 1,
3   "nama": "coba",
4   "email": "coba@coba.com",
5   "dob": "0010-10-09T17:00:00.000+0000"
6 }

```

Akses <http://localhost:8080/getcustomer/1>

localhost:8080/newcustomer?n X localhost:8080/getcustomer/1 X localhost:8080/getcustomer/1 X +

localhost:8080/getcustomer/1

JSON Data Mentah Header

Simpan Salin

```

id: 1
nama: "coba"
email: "coba@coba.com"
dob: "0010-10-09T17:00:00.000+0000"


```




6. Ubah method newcust() untuk dapat menentukan tahun sendiri. Jika tahun tidak ditentukan maka nilai defaultnya adalah tahun 2000.





```

@RequestMapping(value = "/newcustomer", method = RequestMethod.POST)
public Customer newCust(@RequestParam(value="name") String name,
                        @RequestParam(value = "email") String email,
                        @RequestParam(value = "tahun", required = false, defaultValue = "2000")int tahun ) {
    Customer customer = new Customer(name, tanggal: 10, bulan: 10,tahun, email);
    try {
        DatabaseCustomer.addCustomer(customer);
    } catch (Exception ex) {
        ex.getMessage();
        return null;
    }
}

```



POST  http://localhost:8080/newcustomer

Authorization  Headers (3)  Body  Pre-request Script Tests

 form-data  x-www-form-urlencoded  raw  binary

	Key	Value
<input checked="" type="checkbox"/>	name	coba
<input checked="" type="checkbox"/>	email	coba@coba.com
<input checked="" type="checkbox"/>	tahun	2018
	New key	Value

Body Cookies Headers (3) Test Results






Pretty Raw Preview JSON  

```

1 {
2   "id": 1,
3   "nama": "coba",
4   "email": "coba@coba.com",
5   "dob": "2018-10-09T17:00:00.000+0000"
6 }

```

localhost:8080/newcustomer?n X localhost:8080/getcustomer/1 X localhost:8080/getcustomer/1 X

     localhost:8080/getcustomer/1

JSON Data Mentah Header


Simpan Salin



```





id: 1
nama: "coba"
email: "coba@coba.com"
dob: "2018-10-09T17:00:00.000+0000"

```

Jik input tanpa menambahkan tahunnya, maka akan menghasilkan nilai default untuk tahun yaitu 2000.



POST  http://localhost:8080/newcustomer

Authorization  Headers (3) **Body**  Pre-request Script Tests

 form-data  x-www-form-urlencoded  raw  binary

	Key	Value
<input checked="" data-bbox="268 526 288 553" type="checkbox"/>	name	coba
<input checked="" data-bbox="268 586 288 613" type="checkbox"/>	email	coba@coba.com
<input checked="" data-bbox="268 647 288 674" type="checkbox"/>	tahun	
	New key	Value






Body Cookies Headers (3) Test Results

Pretty Raw Preview JSON  

```

1 {
2   "id": 2,
3   "nama": "coba",
4   "email": "coba@coba.com",
5   "dob": "2000-10-09T17:00:00.000+0000"
6 }
```

localhost:8080/newcustomer?n X | localhost:8080/getcustomer/1 X | localhost:8080/getcustomer/2 X

     localhost:8080/getcustomer/2

JSON Data Mentah Header

Simpan Salin

```

id: 2
nama: "coba"
email: "coba@coba.com"
dob: "2000-10-09T17:00:00.000+0000"
```

7. Upload ke Github, Done.