Date : 12/09/2020

Spring Boot 7AM

Mr. RAGHU

---------------------------------------------

Maven Session:

https://www.youtube.com/c/NareshIT/search?query=raghu+maven

All Videos

https://www.youtube.com/watch?v=EA43S5R8LSc&list=PLVlQHNRLflP9XSWeY4x4FLwnL3UOIxnTr

**Lombok API - PART-II**

\*) @RequiredArgsConstructor + @NonNull :-

To create a Parameterized Constructor by using selected variabled.

To select variable apply one annotation : @NonNull

-Ex-

@RequiredArgsConstructor

class Emp {

@NonNull

Integer id;

String name;

}

Generated class:

class Emp {

Integer id;

String name;

Emp(Integer id) {

super();

this.id=id;

}

}

\*) If we apply @NonNull, value must be passed while creating object. It mst not be a null value.

=================================================================================================

Ex#2

package in.nareshit.raghu.model;

import lombok.NonNull;

import lombok.RequiredArgsConstructor;

import lombok.ToString;

@ToString

@RequiredArgsConstructor

public class Product {

@NonNull

private Integer id;

@NonNull

private String code;

}

--Runner class--

package in.nareshit.raghu.runner;

import org.springframework.boot.CommandLineRunner;

import org.springframework.stereotype.Component;

import in.nareshit.raghu.model.Product;

@Component

public class TestData implements CommandLineRunner {

@Override

public void run(String... args) throws Exception {

Product p = new Product(10,"A");

System.out.println(p);

}

}

=========================**\*\*\*Combinations\*\*\***=========================

1) @RequiredArgsConstructor is applied without any variable having @NonNull

then Lombok generated Default constrcutor/zero param

@RequiredArgsConstructor

public class Product {

private Integer id;

private String code;

}

Generated code:

public class Product {

private Integer id;

private String code;

public Product(){

super();

}

}

Ex#2)

@NoArgsConstructor //default const

@RequiredArgsConstructor // 1 params

public class Product {

@NonNull

private Integer id;

private String code;

}

Generated code:

public class Product {

private Integer id;

private String code;

public Product(){

super();

}

public Product(Integer id){

super();

this.id=id;

}

}

Ex#3)

@NoArgsConstructor

@RequiredArgsConstructor

@AllArgsConstructor

public class Product {

@NonNull

private Integer id;

private String code;

}

Generated code:

public class Product {

private Integer id;

private String code;

public Product(){

super();

}

public Product(Integer id){

super();

this.id=id;

}

public Product(Integer id,String code){

super();

this.id=id;

this.code=code;

}

}

Ex#4) Lombok gnerates code based on annotations given which may or may not be valid code

when it gets compiled. Java Compiler will check all code lines after code generation.

@NoArgsConstructor

@RequiredArgsConstructor

public class Product {

private Integer id;

private String code;

}

Generated code:

//2 default constructors --> compile time error ->duplicate methods Product()

public class Product {

private Integer id;

private String code;

public Product(){

super();

}

public Product(){

super();

}

}

Ex#5) If no variables in class @AllArgsConstructor generates default const.

@NoArgsConstructor

@AllArgsConstructor

public class Product {

}

Generated code:

//2 default constructors --> compile time error -> duplicate methods Product()

public class Product {

public Product(){

super();

}

public Product(){

super();

}

}

Ex#6)

@NoArgsConstructor

@RequiredArgsConstructor

@AllArgsConstructor

public class Product {

}

Generated code: //compile time error -> duplicate method Product()

public class Product {

public Product(){

super();

}

public Product(){

super();

}

public Product(){

super();

}

}

Ex#7)

@RequiredArgsConstructor

@AllArgsConstructor

public class Product {

@NonNull

private Integer id;

}

Generated code: // compile time error->duplicate method Product(Integer)

public class Product {

public Product(Integer id){

super();

this.id=id;

}

public Product(Integer id){

super();

this.id=id;

}

}

Ex#8)

@NoArgsConstructor

@RequiredArgsConstructor

public class Product {

}

Generated Code:

Generated code: //compile time error -> duplicate method Product()

public class Product {

public Product(){

super();

}

public Product(){

super();

}

}

========================================================================

@Data

@Data : It is a combination of below annotations

@Setter

@Getter

@RequiredArgsConstructor [If @NoNull exist then param const, else default const]

@EqualsAndHashCode

@ToString

Ex#9)

@Data

public class Product {

private Integer id;

}

\*) Above code generates setId(), getId(), toString() , equals(), hashCode(),

default constructor (@RequiredArgsConstructor).

Ex#10) Only constrcutors

@Data

public class Product {

@NoNull

private Integer id;

}

Generated code:

public class Product {

private Integer id;

public Product(Integer id) {

this.id = id;

}

}

\*\*\*\*\* NOTE \*\*\*\*

@Data uses @RequiredArgsConstructor internally that works

only if no Constructor annotation is provided by programmer.

Ex#11)

@Data

@NoArgsConstructor

public class Product {

@NonNull

private Integer id;

}

Generated code:

public class Product {

private Integer id;

public Product(){

super();

}

}

Ex#12)

@Data

@AllArgsConstructor

public class Product {

@NonNull

private Integer id;

}

\*) If we apply any @\_\_\_\_\_Constructor then @Data will ignore @RequiredArgsConstructor execution

public class Product {

private Integer id;

public Product(Integer id) {

super();

this.id = id;

}

}

\*) @AllArgsConstructor is considered , @RequiredArgsConstructor ignored given by @Data.

----------------------------------------------------------------------------------------