Date : 01-Sep-20

Spring Boot 7AM

Mr. RAGHU

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

hr@nareshit.com

Admin : Srikanth (+91-6302 968 665)

Duration: 3 months (1hr 30Min)

Fee: 5000/- INR

email: javabyraghu@gmail.com

FB: https://www.facebook.com/groups/thejavatemple

PDF Copies:

https://www.mediafire.com/file/w5x9w5vcmkwkkdv/RaghuSirNareshITJavaPdfs.zip/file

Spring Boot Sessions:

https://www.youtube.com/watch?v=xuMyjgqWWVs

https://www.youtube.com/watch?v=pcpaxSRN1tg

https://www.youtube.com/watch?v=1qCgn3SCzhw

Spring Boot Application # 3 types of files

a. Starter class

Spring container

Start application

b. Input File (key-val)

application.properties

application.yml

c. Build Information file

Maven-pom.xml | Gradle-build.gradle

=============== **Spring Configuration** ==================

Spring Boot supports two types of Configurations

a. \*\*\*Annotation Configuration

[\*\*Programmer defined class/no support for pre-defined]

b. Java Configuraion

[\*\*Predefined class/even supports Programmer defined class also]

\*)Configuration file is taken as input by Spring container. So, that container

will do its work.

--------------a. \*\*\***Annotation Configuration**-----------------------

i. StereoType Annotation (5)

@Component : Create object to our class

@Repository : Create object + Db operations + Db Exceptions Handling

@Service : Create object + Transaction Management + cal/operation ..etc

@Controller : Create object + Http Call + MVC + Request Processing..etc

@RestController : Create object + Http call + ReSTful webservices..etc

ii. Data Annotation

@Value : static data, \*\*properties value read, SpEL(Spring Expression)

to variables.

@ConfigurationProperties : Load Multiple key-val from propreties

@Order : Execution/Loading order for same type objects

iii. Link/Wire Annotations

@Autowired : Link two different class object

@Qualifier : Select if multiple child objects found at parent

@Primary : Select one child object as priority.

**Ex:**

@Component : It indicates spring container to create object

=> Write public class

=> non-abstract class and not an interface.

1)

@Component("de")

public class DataExport { }

Spring Container creates object to our class using obj name as de

2)

@Component

public class DataExport { }

Spring Container creates object to our class, object name: dataExport

3)

@Component

public class ProductService { }

Spring Container creates object to our class, object name: \_\_\_\_\_\_\_\_\_\_\_

4) Use @Value -> to provide data to varibles (static, properties data, SpEL)

@Component

public class ProductService {

@Value("NIT")

private String scode;

@Value("996")

private Integer optId;

}

5)

@Component("emutil")

public class EmailUtil {

@Value("Gmail")

private String host;

@Value("586")

private Integer port;

}

create object: emutil (host=Gmail, port=586)

6) To read data from properties file: @Value("${key}")

--application.properties--

# Comment Line

# key=val

code=NIT

version=3.3

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

@Component

public class ProcessMode {

@Value("${code}")

private String pcode;

@Value("${version}")

private Double expVer;

}

--Equal core java meaning(for understanding only)---

ProcessMode processMode = new ProcessMode();

processMode.setCode("NIT");

processMode.setExpVer(3.3);

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

7)

---application.properties--

dc=Oracle

url=jdbc:orcl

un=nit

pwd=srs

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

@Component("con")

public class MyDbConn {

@Value("${dc}")

private String driver;

@Value("${url}")

private String url;

@Value("${un}")

private String username;

@Value("${pwd}")

private String password;

}

Req-> create obj -> con(with properties data)