Date : 07/12/2020
Spring Boot 9AM
(new batch)
Mr. RAGHU (NareshIT)

Srikanth (admin): +91-630 29 68 665
https://www.facebook.com/groups/thejavatemple
email : javabyraghu@gmail.com

Demo#1
https://www.youtube.com/watch?v=L7zUhVLgoBA
Demo#2
https://www.youtube.com/watch?v=oCG4w6Rkcag

Chapter#1 Spring Boot Core

*) Rules to work with Spring Boot (Spring Core)

#1. Our Project is called as Child Project
This is connected to parent Project (Spring Boot)
We can create our Project using either Maven or Gradle.

#2. Spring Boot Application must have 3 types of files.
Those are:

i. Starter class / Main class / Entry class/ Bootstrap class
ii. Input Files (application.propertes - or - application.yml)
iii. Build Information file (Maven-pom.xml, Gradle-build.gradle)

i. Starter class / Main class / Entry class/ Bootstrap class
*) This class behaves as a entry point of execution.
It mainly creates Spring container, by taking inputs from Programmer.

Spring container,
a) Detect (Scan) our classes and create object
b) Provid data to object created
c) Link objects based on relations (HAS-A/Association Mapping)
d) Finally Destory objects (when we stop application)

For this programmer has to give two inputs:
I) Spring Bean(class+rules given by container)
II) Spring Configuration File (XML/Java**/Annotation***)

--Sample code--
```
@SpringBootApplication
public class DemoApp {
  public static void main(String[] args) {
     SpringApplication.run(DemoApp.class,args);
  }
}
```
---

*) Spring container two types:
a) BeanFactory(I) (XML) [Legacy Container]

```
  b) ApplicationContext(I) [XML/Java*/Annotation**]  [new container]

=> Spring Boot uses new Container only 'ApplicationContext'.
 *** Impl class is: AnnotationConfigApplicationContext(C)

================================================================================
*) Spring Annotation Configuration:-
--> Creating Object : StereoType Annotations (5)

a) @Component          = Creating object to our class
b) @Repository         = Creating object + DB Operations
(Insert,Update,Delete...)
c) @Service            = Creating object + Calculations/Operations +
Transaction Management..etc
d) @Controller         = Creating object + HTTP Protocol (web
application)
e) @RestController     = Creating object + HTTP protocol (Global Data
support + REST calls)

--> Data/Link specific

a) @Value -> Provide data
              static data/expression/****data from input files

b) @Autowired / @Qualifier / @Primary
            -> Linking objects

---Examples----------------------
@Component
class DbConn { }
=> Creates object using className first letter small in spring
container
   DbConn dbConn = new DbConn();

@Component("con")
class DbConn { }

=> we can provide object name also,
  DbConn con = new DbConn();

@Component
class DbConn {
  @Value("OracleDriver")
  String driver;
}

=> we can provide data to variables using @Value("Data")
```