```
Profiles in SpringBoot
=> Both Spring and SpringBoot supports profiles
=> In all approaches of Spring app development we can use profiles
     a. xml driven
     b. annotation driven
     c. 100% code configuration
     d. springboot
=>Environment means the setup which is required for executing the app/project
 it contains compiled code + server + dbs/w + jar files + properties/yml etc....
=> From the development to production of the project we come across multiple
environment where code needs to be executed
  for different purpose. This environment is called "Profile".
     a. Dev environment/Dev profile
     b. test environment/Test profile
     c. uat environment/uat profile
     d. prod environment/prod profile
                 etc....
In our SpringApp/project
     a. To make Sterotype annotation based spring class participating only on
certain profiles activation we use
                 "@Profile" on top of Spring bean class like @Profile("dev") or
@Profile({"dev", "test"})
     note: if no @Profile is specified the spring bean is available for all the
profiles.
     @Repository("mySQLEmployeeDAO")
     @Profile({"dev", "test"})
     public class MySqlEmployeeDaoImpl implements IEmployeeDAO{
     }
     @Repository("oracleEmployeeDAO")
     @Profile({"uat", "prod"})
     public class OracleEmployeeDaoImpl implements IEmployeeDAO{
     }
     b. To make @Bean method of @Configuration class returned a java class object
which represents a sprint bean working on certain profiles
        then @Profile annotation should be used on top of @Bean method
        @Configuration
        public class PersistenceConfig{
           @Bean("dsDBCP2")
           @Profile("dev")
           public DataSource createUsingDBCP2(){
           }
           @Bean("ds0raUCP")
           @Profile("dev")
           public DataSource createUsingOracleUCP(){
           }
```

```
=> To keep application.properties/yml file into specific one profile, we create
seperate properties file for that profile as shown below
                  syntax:: application-ofile>.properties/yml
                  application.properties
                  application-dev.properties/yml
                  application-uat.properties/yml
                  application-pro.properties/yml
                  application-dev.properties/yml
How to activate the specific profile
      a. In application.properites/yml
                  spring.profiles.active = dev
      b. using -Dspring.profiles.active = dev(system property with value)
Note: if any give profile related properties file is not available then it will use
application.properties file as fallbackproperties file.
=> Getting employee record on the basis of Designation... use V0, B0, DT0
EMPDTO
=====
empNo
empName
iob
sal
deptNo
MgrNo
controller => List<EmployeeVO>showEmpsByDesg(String[] desg)
           => List<EmployeeDTO>fetchEmpsByDesg(String[] desg)
service
dao
           => List<EmployeeB0> getEmpsByDesg(String cond)
sequence:: client<======V0======controller<=====DT0=====> service<=====B0======
DA0
Tommo
=====
yml usage and benefits in profiles
Commandline Runners
Till monday
========
SpringJDBC, SpringORM, SpringDataJPA
SpringBootMongoDB, SpringMVC(introduction)
recorded videos
===========
Maven, Junit, lombok, log4j, SpringBootlogging(videos)
```