Date : 30/12/2020
Spring Boot 9AM
Mr. RAGHU

Spring Boot Profiles : Using @Profile Annotation

*) @Profile annotation is used to executed logic (code) based on Environment provided.

--Example--
```
package in.nareshit.raghu.runner;

import org.springframework.boot.CommandLineRunner;
import org.springframework.context.annotation.Profile;
import org.springframework.stereotype.Component;

@Component
//@Profile("prod")
@Profile({"prod","qa"})
public class EmailAlert implements CommandLineRunner {

        @Override
        public void run(String... args) throws Exception {
                System.out.println("FROM EMAIL ALERT");
        }
}
```

*) active using
```
   --spring.profiles.active=prod  (or)
   --spring.profiles.active=qa
```

Q) Do we need to have properties file for current profile?
A) No Need. Consider your current profiles is prod,
   then we can create application-prod.properties with keys.
   If we do not create application-prod.properties
   or else file created without keys then then Fallback is implemnted
   ie read data from application.properties.

   ** Profile based properties files are optional.

-----new code------------
a) Runner class
```
package in.nareshit.raghu.runner;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.boot.CommandLineRunner;
import org.springframework.context.annotation.Profile;
import org.springframework.stereotype.Component;

@Component
//@Profile("prod")
@Profile({"prod","qa"})
public class EmailAlert implements CommandLineRunner {
        @Value("${my.app.title}")
        private String title;

        @Override
        public void run(String... args) throws Exception {
                System.out.println("FROM EMAIL ALERT => " + title);
        }
}

b)
--application.properties--
my.app.title=DEFAULT

--application-prod.properties--
my.app.title=PRD_TITLE

--application-qa.properties--
my.app.title=QA_TITLE
___________________________________________________________________
Autowired:
https://www.youtube.com/watch?v=-
FlszP92JVM&list=PLVlQHNRLflP9XSWeY4x4FLwnL3UOIxnTr&index=13

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


--> Fn + F4 : TO see sub class/ interface impl classes.

  ---------------------------------------------------------------------
----
                @Profile + service interface + properties
  ---------------------------------------------------------------------
----

#1 Create Starter Project
Name : SpringBoot2ProfileAnnoTwo
Dep : lombok

#2. Service Interface
package in.nareshit.raghu.service;
public interface AlertService {
        public void showAlertMsg();
}

#3. Service Impls
package in.nareshit.raghu.service.impl;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Profile;
import org.springframework.stereotype.Component;

import in.nareshit.raghu.service.AlertService;

@Component
@Profile("default")
```

```
//@Profile({"default","dev"})
public class SmsAlertService implements AlertService {

	@Value("${my.app.service.code}")
	private String scode;

	@Override
	public void showAlertMsg() {
		System.out.println("FROM MESSAGE SERVICE => " +
scode);
	}

}
---
package in.nareshit.raghu.service.impl;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Profile;
import org.springframework.stereotype.Component;

import in.nareshit.raghu.service.AlertService;

@Component
@Profile({"qa","prod"})
public class EmailAlertService implements AlertService {

	@Value("${my.app.service.code}")
	private String scode;

	@Override
	public void showAlertMsg() {
		System.out.println("FROM MAIL SERVICE => " + scode);
	}

}
---------
#4 properties files

--application.properties--
my.app.service.code=SMS-SERVICE

---application-qa.properties--
my.app.service.code=EMAIL-qa-SERVICE

---application-prod.properties--
my.app.service.code=EMAIL-prod-SERVICE

-----------------------
#5. Runner class
package in.nareshit.raghu.runner;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.stereotype.Component;

import in.nareshit.raghu.service.AlertService;
```

```
@Component
public class ExecuteServiceRunner implements CommandLineRunner {
        /**
         * -> goto container,
         * -> read impl class object
         * -> inject object to this(service) variable
         */
        @Autowired
        private AlertService service;

        @Override
        public void run(String... args) throws Exception {
                service.showAlertMsg();
        }

}
```

Q) How can we activate profile?
A) --spring.profiles.active=name

@Profile("name") -->code

        *** Link For Latest Modifications on Profiles ****
https://spring.io/blog/2020/08/14/config-file-processing-in-spring-boot-2-4

In version 2.4.x we want to work as 2.3.x or before concpts.
Then add :

```
    spring.config.use-legacy-processing=true
```

inside application.properties/ application.yml

----

a. Profiles properties --> Env based loading
b. --spring.profiles.active=name
c. If keys not present in current profile then fallback -> application.properties
d. One profile properties we can divide into small properties file finally incude them
   --spring.profiles.include=prodemail,prodb,prodsecurity
e. @Profile annotation for code/logic execution based on environment
f. we can mix @Profile + propreties concept.
   Properties for profile is optional (fallback is appiled).
g. If no profile is active, then default is active.
   [application.properties]
h. Incase of YAML we can use single file for multiple profiles
   using 3 dash symbols (not supported child profiles, if you want write
   individual YAML files).

```
 ______application.yml_____
 key: val
 ---
 spring:
   profiles: <current_profile_name>
 key: val
 ---
 spring:
   profiles: <current_profile_name>
 key: val
```
____

```
Q) If both YAML and Properties files exist then which one is loaded?
A)
  Spring Boot 2.4.x -- YAML is priority
  Spring Boot 2.3.x -- properties is priority
----------------------------------------------------------------------
------
```