Day 3 – Spring Boot Application, Spring cloud Setup

Id:[kunalmittal.houston@gmail.com](mailto:kunalmittal.houston@gmail.com)

Password: Miracle@123

1. By annotating MockMailSender with @Component, we told Spring to create an instance of MockMailSender and keep it at "some place" when the application starts.
2. By annotating the mailSender variable with @Autowired, we told Spring to go to "that place," search for an object of type MailSender, and assign a reference of that to this variable.

That place, where Spring kept the MockMailSender instance, is called **Application Context**.

Annotations :

Spring will create instance of the classes annotated with following keyword in application context container .

Spring creates beans for classes annotated with @Component or one of its specializations, e.g.:

* @RestController
* @Controller
* @Service
* @Repository
* @Configuration
* @SpringBootApplication

In fact, we have already seen @Component and @RestController in action.

**private** MailSender mailSender;

**public** MailController(@Qualifier("smtpMailSender") MailSender smtp) {

**this**.mailSender = smtp;

}

@RequestMapping("/mail")

**public** String mail() {

mailSender.send("mail@example.com", "A test mail", "Body of the test mail");

**return** "Mail sent";

}

////////////////////////////////////////////////////////

//@Autowired

**private** MailSender mailS ;

//private MailSender mailsender = new MockMailSender();

//@Autowired

//@Autowired

**public** MailController( @Qualifier("smtpMail") MailSender smtp

) {

**this**.mailS = smtp ;

}

@RequestMapping("/mail")

**public** String mail() {

mailS.send("Mail@example.com", "Hello this is the test mail", "body of the test mail");

**return** "Mail Sent..vzxcvz." ;

}

//////////////////////////////////////

Configuration file

///////////////////////////////////////////////////

So, to summarize, Spring maintains an Application Context where it keeps some objects, which are also called **beans**. When a Spring application starts, Spring looks for some metadata — like the @Component annotation — for building the application context. The @RestController annotation, which our MailController is annotated with, is also like the @Component annotation — Spring also creates instances of classes annotated with @RestController. That’s why when our application started, Spring created a MailController bean as well. That time, seeing that its mailSender field was annotated with the @Autowired annotation, Spring assigned mailSender a reference to the MockMailSender bean that it had already stored in the application context

So, when developing a Spring application, you basically have to know two things:

1. How to tell Spring to put objects into the application context. i.e. what kind of metadata, like the @Component annotation, to use.
2. How to tell Spring to inject in your code references to the objects in the application context, i.e. what kind of metadata, like the @Autowired annotation, to use.

The IoC container is responsible to instantiate, configure and assemble the objects. The IoC container gets informations from the XML file and works accordingly. The main tasks performed by IoC container are:

* to instantiate the application class
* to configure the object
* to assemble the dependencies between the objects