**Spring @ConditionalOnProperty @Order- 2022**

The annotation is used to conditionally create a Spring bean depending on the configuration of a property.

**public interface Notifier {**

**void notify(String message); 🡸 Interface**

**}**

**public** **class** EmailNotifier **implements** Notifier {

@Override

**public** **void** notify(String message) {

System.***out***.println("Sending email message: "+message);

}

}

**public** **class** SMSNotifier **implements** Notifier {

@Override

**public** **void** notify(String message) {

System.***out***.println("Sending SMS message: "+message);

}

}

**Configuration to create bean based upon the value in application.properties**

@Configuration

**public** **class** NotificationConfig {

@Bean(name="notifyByEmail")

@ConditionalOnProperty(name="service.type",havingValue = "email")

**public** Notifier emailNotifier() {

**return** **new** EmailNotifier();

}

@Bean(name="notifyBySMS")

@ConditionalOnProperty(name="service.type",havingValue = "sms")

**public** Notifier smsNotifier() {

**return** **new** SMSNotifier();

}

}

@Component

**public** **class** AutoRun {

@Autowired

**private** Notifier notifier;

@EventListener(ApplicationReadyEvent.**class**)

**public** **void** run() {

notifier.notify("Some message");

}

}

**application.properties**

#service.type = email

service.type = sms