**Externalized Configuration**

Using-

1. properties files
2. YAML files
3. environment variables
4. command-line arguments

Property values can be injected directly into your beans by using the @Value annotation

Every enterprise application has many environments, like:

**Dev | Test | Stage | Prod | UAT / Pre-Prod**

Each environment requires a setting that is specific to them. We make properties files for each environment and set the profile in the application accordingly, so it will pick the respective properties file

as we create

1. application-dev.properties
2. application-test.properties
3. application-prod.properties

spring.profiles.active=dev

spring.application.name=employeeservice

app.message=This is the primary Application Property for ${spring.application.name}

After  properties files. DatabaseConfiguration.class configure  to pick the correct one.

We have used the @Profile(“Dev”) to let the system know that this is the BEAN that should be picked up when we set the application profile to DEV. The other two beans will not be created at all.

Then set spring.profiles.active=dev in property file to let the system know that this is DEV, TEST, or PROD. Spring Boot will know which profile to pick.

From command line use------mvn spring-boot:run -Dspring-boot.run.profiles=test