











## Using @Value annotation

* Can access values from application.properties file by key
* Here the my.fullGreetDesc is injected into greetMsg
* Code
* *@RestController*
* public class Controller {
* *@Value*("${my.fullGreetDesc}")
* private String greetMsg;
* *@GetMapping*("/greet")
* public String getGreeting() {
* return greetMsg;
* }
* }

## Tricks while using @Value

1. If suppose key -value is not present in application.properties the which try to access using @Value, then in that case application run into error.
   * To avoid that @Value(“${my.greet:default value}”) => in this it will print default value instead of error
2. Suppose we got list as comma separated in app.prop file

my.list=one, two, three

In order to access as List<String> directly we can make it as

@Value(my.list)

List<String> list; //=> one,two, three populate as list

## Using @ConfigurationProperties

Suppose I want get value of key which associate with db from app.properties

db.connection={connectionString:'http://....', userName:'foo', password:'pass'}

db.host=127.0.0.1

db.port=1200

*@Configuration*

*@ConfigurationProperties*("db")

public class DbConnection {

private String connection;

private String host;

private int port;

public String getConnection() {

return connection;

}

public void setConnection(String connection) {

this.connection = connection;

}

public String getHost() {

return host;

}

public void setHost(String host) {

this.host = host;

}

public int getPort() {

return port;

}

public void setPort(int port) {

this.port = port;

}

}

*@Autowired*

private DbConnection dbConnection;

*@GetMapping*("/db")

public String getDbDetails() {

return dbConnection.getConnection()+dbConnection.getHost()+dbConnection.getPort();

}

## Running jar file from cmd line

* mvn -v => check if maven installed
* mvn install
  + in target folder jar file gets created
* java -jar <name of jar file>

## Externalize the properties file

* Scenario
  + Suppose we create a jar file of application, we no longer able access application.properties file of application
  + In that case what we create a property file at target file. Now we can change value of any key externally and the application automatically pick the value from external property file instead of one which is within jar file
  + We can also pass value to key through command line as well
* In target folder run
  + Method 1
    - *vi application.properties* => in linux
    - *notepad application.properties* => in windows
    - it create application.properties file and opens it
    - Now if we run jar file it detects application.properties file from target file and use it in application instead of one which id within application
  + Method 2
    - Pass it through cmd itself
    - *java -jar <jar file name> --<key>={value we want to pass}*
      * ex :- java -jar spring-boot-config-0.0.1-SNAPSHOT.jar --my.fullGreetDesc="from cmd"

\*\* **What if we use both methods together?**

* It will first search for application.properties file from jar, them from target file. If it is there it overrides. And further if it is passed through cmd it overrides with cmd value.