14-2-22

Spring Boot

What is spring boot? How it is different from spring

If we need mvc in spring we need an external server to run it like tomcat or any other server

In spring boot we can run both application one is simple standalone and second one is our web application

In web application we can create the websites or any web services

In spring boot we don’t need xml file for configuration

We removed it and we use all of annotations to configure the application.

@autowired, @component, @componentScan, @Bean

In spring there will be one main file which will load the application

Support constructor binding for property nested inside a javabean

Lazy initialization:- @Lazy in our spring core annotation

Before this spring boot also support now java 13 last supported version was 11

Immutable binding:- @ConfigurationProperties

@ConstructorBinding :- it helps us in immutable binding of

Spring Security :- RSocket Support.

The logging file in spring boot is now renamed to logging.file.name

This is change from **Logging.path ->logging.file.path**

WE first see the difference between our spring boot and spring Core:-

|  |  |
| --- | --- |
| Spring boot | Spring core |
| It is mainly used to develop the REST API | It is widely used to develop the java EE Application |
| To short he length of code and provides the easy way to develop the web applications. | With spring core we can simply the java EE Application development so that our developers are more productive |
| Main focus is on autoconfiguration. It always auto configure the classes based on the requirement. | Main feature is dependency injection |
| It helps to create a stand alone application. | In spring core we got the loosely coupled applications. |
| As it focus on auto configuration, need of writing lots of code is removed. | We need to write so many code. First of all we need to create the class, create dependent class. After this step we move to xml file and declare the beans. It means we write so many of the codes. |
| It has inbuilt application server | We need the server only when we required it like in mvc |
| As it has a inbuilt database support | We need to connect to database when we are developing the application |

Spring boot vs spring MVC

**Spring Boot Architecture :-**

**Spring boot can be connected from any front end like angular, react or vue js**

Presentation layer ( Authentication & JSON Tranlation)

1. Business logic
2. Validation
3. authorization

Business Layer

Persistence Layer :- Storage Logic

Actual Database:- Actual Database

Crud Services by Repository

Spring Boot Flow

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Client |  | Controller |  | Service Layer |  | Model |

Actual Database

DAO Class

@RestController

@RequestMapping(“/”)

@RestController

Class RSTController

{

@RequestMapping(“/hmsg”)

public String hello()

{

return “hello how are you!!!”;

}

}

Spring boot provides direct connection to different databases:- it means we just need to provide connection url and it is going to provide us the object for query executing

We need spring jdbc connection

And second database connector in our case we are using mysql as database so we need mysql connector for spring boot

First steps is provide connection url in application.context file

Second provide JDBCTemplate as autowired in any of our file

With Rest APi we get the data from different sources. So to make them compatible we use json format or object to insert, update, delete and select

#mysql db-“emp”

All code to connection for mysql will be here

#oracle db=”student”

We will write all code of connection here

Com.yash

-emp

-student

Date :15-2-22

Thymeleaf:-

Xml, valid xml, xhtml, valid xhtml, html5, older version of html is also supported

Standard Dialect

Where a object is applies to DOM, and set of processors along with some the extra artifacts.

We can use the thymeleaf

Steps to create a form with thymeleaf

1. Create a bean class which will work as model with form data
2. Create a controller class, in controller class we will submit the form and values will be passed.
3. We create a template file inside src/main/resource/templates/ all html files will be here like we will put our login.html, signup.html
4. We need to put enteries in our application.properties file for thymeleaf
5. Spring.thymeleaf.cache:false
6. Spring.thymeleaf.suffix:.html
7. Suppose the model name is user/customer/client
8. <input type=”text” name=”userfname” value=”default value” th:value=”${user.userfname}” />

Div, div creates the division in page. Span is same tag but having no line break.

JPA :- Java Persistant Api

We can use the jpa to manage the data between database and classes

Jpa follows the ORM.

JPA:- Entity Manager API for processing the queries and transections.

It is an set of interfaces

When we use the jdbc and hand written sql we need to write lots of code

JPA POJO to represent persistent data .

Jpa over the plain JDBCTemplate

1. Avoid writing the ddl in database specific. It maps the request from xml or using java annotations.
2. Jpa also avoid writing the DML command for database specifics.
3. We will use here JPQL to run the query over the java entities rather than the sql tables and columns.

JPA Architecture:-

Persistence:- it is a class that contain static methods to obtain an entitymanagerfactory object.

EntittyManagerFactory:- it is a factory class of EntityManager it creates and manage multiple instatnce of EM.

EntityManager is an interface which controls the persistence operations on objects.

Entity:- the entities are the persistence objects stores as a record in the database. :- I say entity is replica of table that we have created / or created by JPA

Persistence Unit:- so many tables, we can have so many entities like user, city, job profile, products

EntityTransaction:- it mapped with entity manager by one to one mapping

Query :- it is an interface which use to provides us the result. We can write here our own functions to generate the result as per our need.

JPA Class Relationship

EntityTransaction

EntityManagerFactory

Query

\*

1

\*

EntityManager

Persistence

\*

@Entity:- any class marked with @Entity it become the table in the database

@Id is use to mark a column as primary key in side our entity class.

From now what we are going to do

We are going to create different layers in our project

Say we will put all of our entity in model package.

We will put all of our controller in controller package.

We are going to create a service layer. And it will be in our service package. For each entity we will have different service classes. Like for user entity we will haver userService class and for product we will have productService.

For repository we are going to create a layer with repository name

All of the implementation for entity will be here

Like we are having product, user, and other entity so in that case we will have a

UserRepository , ProductRepository.

Com.yash.UMS

When we pass data from form :- it will populate the bean/model /Entity for storing in database.

As we are using service layer we are going to call the service methods in our controller

UserService

@Service

Spring Junit Test

You already know about the junit which is used for automated testing in java

So we can test our spring application using junit

Each and every part of spring application can be tested with the help of unit testing

We required all the jar files(maven repository is going to be called ) like spring boot, jdbc, jpa, Jupiter, thymleaf

The way develop our application we are going to develop in the same sequence but with the integration of the our junit testing.

When we say httprequest test it means we are going to test the application with url

http://localhost:81/show