Phase 3

9 classes

Day 4 : June 11 2024

SQL

Select \* from employee; here employee is table. SQL is not case sensitive. \* means retrieve all records with all columns. We get output as string format. In DAO we need to convert into objects.

Where clause

Select \* from employee where id=100;

Select \* from employee where salary > 25000

Retrieve specific column like id or name or name and salary

Select id from employee; retrieve all ids.

Select name from employee retrieve all names

Select name, salary from employee retrieve name and salary

JPQL (Java Persistence Query Language)

Select emp from Employee emp; here Employee is entity class and emp is object name. case sensitive. emp retrieve all variable value from employee table.

Select emp from Employee emp where emp.id=100; here emp is object and id is variable name.

Select emp from Employee emp where emp.salary > 25000

Retrieve partial object.

Select emp.id from Employee emp;

Select emp.name from Employee emp;

Select emp.name, emp.salary from Employee emp;

Spring Framework provide IOC and DI features

JPA or Hibernate provide ORM

Spring ORM module help to integrate with existing ORM tool like Hibernate or JPA.

Spring MVC :

Spring MVC is another module provided by spring framework which internally follow MVC architecture.

Limitation of Servlet.

If we want to create Servlet class we need to create normal java class and that class must be implements or extends Type of servlet. Then we need to override pre defined method like doGet, doPost, doPut or doDelete. If we want to use more than one doGet method inside one servlet not possible.

If we want to create Servlet concept using Spring MVC.

@Controller it is like a Servlet.

class MyController {

then we can write more than one user defined method as get, post , put or delete. Method return type must be ModelAndView or String with few configuration in xml file.

@RequestMapping(value=”hello”,method=RequestMethod.GET)

public ModelAndView sayHello() {

// coding ….

ModelAndView mav = new ModelAndView();

mav.setViewName(“display1.jsp”); // like forward

return mav;

}

@RequestMapping(value=”hi”,method=RequestMethod.GET)

public ModelAndView sayHi() {

// coding ….

ModelAndView mav = new ModelAndView();

mav.setViewName(“display2.jsp”); // like forward

return mav;

}

}

In Spring MVC project we need to configure front controller pre defined class is DispatcherServlet.

Spring MVC internally follow front controller design pattern.

According to Front controller design pattern whenever we want to pass any request it must be pass through servlet class and servlet class keep the track about all the request.

This front controller class we need to configure in web.xml file or separate configuration class.

<servlet>

<servlet-name>dispatcher</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

Whenever any request (/) receive from client application url pattern pass that request to DispatherServlet class. DispatcherServlet is a pre defined class provided by spring framework it behave like FrontController.

Then DispatcherServlet search spring configuration file start with pre-fix as servletname-servlet.xml. dispatcher name part of servlet-name in xml must be match with xml configuration file name.

If servlet-name tags value is dispatcher then file name must be dispatcher-servlet.xml

If servlet-name tags value is Abc then file name must be Abc-servlet.xml

If servlet-name tags value is obj then file name must be obj-servlet.xml

Index.jsp (first page)--🡪 hyperlink hello 🡪 once you click on hyperlink the request pass to web.xml file. In Web.xml file we are mapping url-pattern as /. Means any request come from view it can receive by DispatcherServlet. Then this class search spring configuration file start with pre-fix servletname-servlet.xml. Inside this file we can do some configuration depending upon our project requirement. As of now we enable @Controller annotation with the context:component-scan tag. Spring container scan com package inside that package with @Component, @Controller, @Service as well as @Repository annotation classed do the DI. Then inside @Controller class it check @RequestMapping annotation value. If value match it invoke that method and do the coding and re-direct to specific view page part of ModeAndView class reference.

Spring MVC with ORM tool as JPA