org.apache.maven

webapp

name it as SpringMVC

source folders

to get it go to built path order and select on folders

go to maven repository

Servlet

<!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

Web.xml we are going to

Deployment descriptor archtypes reated web application

version 2.3 is default

during the deployement it gives the exception for that we need to change it to 3.1 or 4

delete web.xml -> write click on project sho in -> system explored

setting

there wil be only one xml file

open

change the virsion as 3.1

update the project

-----------------

go to project right click java tools genereate web.xml

-------------------------------------------------------------

WebXML congiration

<web-app>

<Servlet-name> Dispatcher</Servlet-name> //we can give any name

<Servlet-class>DispatcherServlet<Servlet-class> //DispatcherServlet is class it comes from spring core

</web-app>

1) Spring core

5.2.9.Reliea

<!-- https://mvnrepository.com/artifact/org.springframework/spring-core -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.3.9</version>

</dependency>

2) Sprint context

5.2.9

<!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.9</version>

</dependency>

LomBok

<!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.12</version>

<scope>provided</scope>

</dependency>

Spring AOP

-----------------------

SpringWebMvc instead all above dependency we can put dependecy SpringWebMvc then it will add all the dependency which required for the dynamic project

<!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.2.9.RELEASE</version>

</dependency>

---------------------------------------------------------------------------------------------------------

inside web-inf folder

create dispatcher.xml name should be same to provide the configuration

we will write all java programs in src/main/java

create package

com.te.springmvc .................. consider this is base package

just create class so that package does not disappear

create new sub package

1. constroller --------module name is constroller for our understanding

2. create class SpringMVCConstroller class

it will become controller

when we annotate class with @controller

@controller

public class SpringMVCcomponent{

//inside this class we will write handler methods

@RequestMapping(path="/home")

public String xxx(){

return home.jsp

}

}

@constroller and @component does the same job we can use any of those

------------------------------------------------------------

create folder inside the WEB-INF/Views

inside this create all the jsp pages

always path as /WEB-INF/VIEWS/jsppage.jsp

when you are using @constroller [or] @component to the class to make it eligible for instantiation then it is mandatory to configure dispatcher.xml to configure with

<context-compononetScan : base-package="">

request will be given

to send the data from controller to jsp

There are three ways

1. Model iterface using mathod there is something knows model and view class

methd addAttribute()

2. ModelView

Method addObject()

3. ModelMap

addAttribute(ModelMap model)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*frist way

create new sub package

1. constroller --------module name is constroller for our understanding

2. create class SpringMVCConstroller class

it will become controller

when we annotate class with @controller

@controller

public class SpringMVCcomponent{

//inside this class we will write handler methods

@RequestMapping(path="/home")

public String xxx(Model model){ //Model is interface in which addAttribute method is present.....We are passing the date in key and value format

model.addattribute("name", "Shubham")

return "home.jsp";

}

}

--------------------------------------------------

in JSP pages

<%

String string = request.getAttribute ("name") ; //it accepts the key which we have passed in the addAttribute() method so it will return the value "Shubham" which we have passsed there

%> // we got the data now but we are not printing

<html>

<h1> Welcome <%=string%> </h2>

<html>

===================================================================================================

Secong View

create new sub package

1. constroller --------module name is constroller for our understanding

2. create class SpringMVCConstroller class

it will become controller

when we annotate class with @controller

@controller

public class SpringMVCcomponent{

//inside this class we will write handler methods

@RequestMapping(path="/home")

public String xxx(ModelAndView model){ //ModelAndView is the class , it consist of both model name and view name

model.setViewName(home.jsp); //we need to set the view using this method and return ModelAndView reference only other wise it returns null;

model.addObject("name", "Shubham")

return model;

}

}

--------------------------------------------------------

Program 2

Creating Views

loginForm.jsp

---------------------------------------------------------------

4 ways to get the data from a formalpara

Httpsservlet request

request.getparameter();

=-------------------------------------------------------------------------------

\*\*\*\* 1 st way of gettign

(dot) represents current project

------------------------------------------------------------------

Secong way of getting the form data @Requestparam

Public String getFormDatatwo(ModelMap map,@ReqestParam(“empID”) int id), @RequestParam(“pass”) String pass )

{

Map.addAttribute(“empid”,id);

Map.addAttribute(“pass”,pass);

}

//empId, pass is coming from the web browser

Public String getFormDatatwo(ModelMap map,@ReqestParam int id), @RequestParam String pass )

{

Map.addAttribute(“empid”,id);

Map.addAttribute(“pass”,pass);

}