**Custom Dispatcher Servlet And Image store processing**

In this we are going to discuss How to use Custom Dispatcher Servlet and How to use Multipart Request ( image, pdf, doc etc files process) in Spring MVC.

* Add new Servlet in your Application we can add a new servlet also in our application

By overriding method of AbstractAnnotationConfigDispatcherServletInitializer

This class already implemented DS.

1. If we want to process image then we will have use multipart request.

And we have to add MultipartResolver to display multipart request( pdf, image, doc etc) otherwise will not display.

@Bean

**public** MultipartResolver multipartResolver() **throws** IOException {

**return** **new** StandardServletMultipartResolver();

}

1. Add one more tag **enctype** with value **multipart/form-data**

<form:form action=*"dojstudent"* commandName=*"student"* enctype=*"multipart/form-data"*>

<table style="text-align: *center*; position: *absolute*;padding-left: *500px*;">

<tr>

<td>First Name : </td>

<td><input name=*"fname"* type=*"text"*></td>

<td><form:errors path=*"fname"* cssClass=*"error"*/></td>

</tr>

1. We have to use @RequestPart to handle multipart request data.

@RequestMapping(value="/dojstudent", method=RequestMethod.***POST***)

**public** String getStudent(ModelMap model, @Valid Student student, Errors errors, @RequestPart("profilePicture") byte[] profilePicture){

**if**(errors.hasErrors()){

System.***out***.println("Errors "+errors.getErrorCount());

**return** "student";

}

In @RequestPart is handling by name profilePicture then byte stream store reads in bytes and resolve it by the help of multipart resolver.

profilePicture as request part which comes in byte stream array. This byte stream is reading and resolve by multipart resolver as file and save it. But file is temporary save and unreadable file.

We can avoid this by using Multipart file as a request.

Like:

@RequestMapping(value="/dojstudent", method=RequestMethod.***POST***)

**public** String getStudent(ModelMap model, @Valid Student student, Errors errors, @RequestPart("profilePicture") MultipartFile profilePicture){

**if**(errors.hasErrors()){

System.***out***.println("Errors "+errors.getErrorCount());

**return** "student";

}

**try** {

profilePicture.transferTo(**new** File("E:\\Shiksha\\Programming\\DOJWorkspace\\"+profilePicture.getOriginalFilename()));

} **catch** (IllegalStateException | IOException e) {

e.printStackTrace();

}

@Override

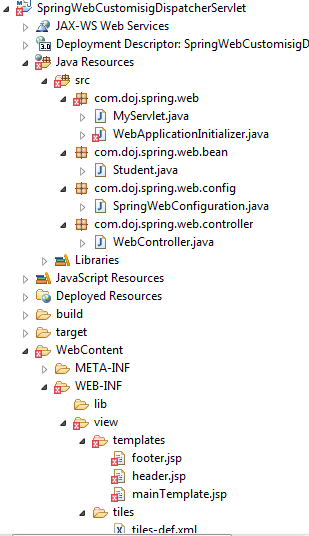
**public** **void** onStartup(ServletContext servletContext) **throws** ServletException

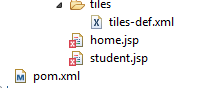
{

Dynamic myServlet = servletContext.addServlet("myServlet", MyServlet.**class**);

myServlet.addMapping("/custom-servlet/");

}





**package** com.doj.spring.web;

**import** javax.servlet.http.HttpServlet;

**public** **final** **class** MyServlet **extends** HttpServlet {

/\*\*

\*

\*/

**private** **static** **final** **long** ***serialVersionUID*** = 3347800094882299611L;

}

**package** com.doj.spring.web;

**import** javax.servlet.MultipartConfigElement;

**import** javax.servlet.ServletRegistration.Dynamic;

**import** org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

**import** com.doj.spring.web.config.SpringWebConfiguration;

//this file is equivalent to web.xml

**public** **class** WebApplicationInitializer **extends** AbstractAnnotationConfigDispatcherServletInitializer{

//Configuration for non web components like services, daos, repos, etc.

@Override

**protected** Class<?>[] getRootConfigClasses() {

**return** **null**;

}

//Specifying Spring MVC configuration class "SpringWebConfiguration.class" it equivalent to \*-servlet.xml file

@Override

**protected** Class<?>[] getServletConfigClasses() {

**return** **new** Class<?>[]{SpringWebConfiguration.**class**};

}

//Mapping dispatcher server to "/" i.e. Servlet Mapping in the web.xml

@Override

**protected** String[] getServletMappings() {

**return** **new** String[]{"/"};

}

/\*@Override

public void onStartup(ServletContext servletContext)

throws ServletException {

Dynamic myServlet = servletContext.addServlet("myServlet", MyServlet.class);

myServlet.addMapping("/custom-servlet/");

}\*/

@Override

**protected** **void** customizeRegistration(Dynamic registration) {

registration.setMultipartConfig(

**new** MultipartConfigElement("E:\\Shiksha\\Programming\\DOJWorkspace", 204834l, 4937353l,0));

}

}

**package** com.doj.spring.web.bean;

**import** javax.validation.constraints.Max;

**import** javax.validation.constraints.Min;

**import** javax.validation.constraints.NotNull;

**import** javax.validation.constraints.Size;

**import** org.hibernate.validator.constraints.NotEmpty;

**public** **class** Student {

@NotNull

@NotEmpty(message="Your first name is mandatory")

@Size(max=20, min=5, message="Your first name must be between 5 to 20 characters")

String fname;

@NotNull

@NotEmpty(message="Your last name is mandatory")

@Size(max=20, min=5, message="Your last name must be between 5 to 20 characters")

String lname;

@NotNull

@NotEmpty(message="Your address is mandatory")

@Size(max=50, min=10, message="Your address must be between 10 to 50 characters")

String address;

@NotNull

@NotEmpty(message="Your course is mandatory")

@Size(max=10, min=3, message="Your course must be between 3 to 10 characters")

String course;

@NotNull

@NotEmpty(message="Your age is mandatory")

@Max(value=50, message="Your age should be less than or equal to 50")

@Min(value=5, message="Your age should be greater than or equal to 5")

String age;

**public** String getFname() {

**return** fname;

}

**public** **void** setFname(String fname) {

**this**.fname = fname;

}

**public** String getLname() {

**return** lname;

}

**public** **void** setLname(String lname) {

**this**.lname = lname;

}

**public** String getAddress() {

**return** address;

}

**public** **void** setAddress(String address) {

**this**.address = address;

}

**public** String getCourse() {

**return** course;

}

**public** **void** setCourse(String course) {

**this**.course = course;

}

**public** String getAge() {

**return** age;

}

**public** **void** setAge(String age) {

**this**.age = age;

}

}

**package** com.doj.spring.web.config;

**import** java.io.IOException;

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.ComponentScan;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.web.multipart.MultipartResolver;

**import** org.springframework.web.multipart.support.StandardServletMultipartResolver;

**import** org.springframework.web.servlet.ViewResolver;

**import** org.springframework.web.servlet.config.annotation.DefaultServletHandlerConfigurer;

**import** org.springframework.web.servlet.config.annotation.EnableWebMvc;

**import** org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;

**import** org.springframework.web.servlet.view.tiles3.TilesConfigurer;

**import** org.springframework.web.servlet.view.tiles3.TilesViewResolver;

@Configuration

@EnableWebMvc

@ComponentScan("com.doj.spring.web.controller")

**public** **class** SpringWebConfiguration **extends** WebMvcConfigurerAdapter{

//Apache tiles

//Configuring and creating bean for view resolver

/\*@Bean

public ViewResolver viewResolver() {

InternalResourceViewResolver resolver = new InternalResourceViewResolver();

resolver.setPrefix("/WEB-INF/view/");

resolver.setSuffix(".jsp");

resolver.setExposeContextBeansAsAttributes(true);

return resolver;

}\*/

@Bean

**public** ViewResolver viewResolver() {

**return** **new** TilesViewResolver();

}

@Bean

**public** TilesConfigurer tilesConfigurer() {

TilesConfigurer tiles = **new** TilesConfigurer();

tiles.setDefinitions(**new** String[] {

"/WEB-INF/view/tiles/tiles-def.xml"

});

tiles.setCheckRefresh(**true**);

**return** tiles;

}

//Configure for default static content handling

@Override

**public** **void** configureDefaultServletHandling(DefaultServletHandlerConfigurer configurer) {

configurer.enable();

}

@Bean

**public** MultipartResolver multipartResolver() **throws** IOException {

**return** **new** StandardServletMultipartResolver();

}

}

**package** com.doj.spring.web.controller;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.HashMap;

**import** java.util.Map;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.validation.Valid;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.ui.ModelMap;

**import** org.springframework.validation.Errors;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RequestMethod;

**import** org.springframework.web.bind.annotation.RequestParam;

**import** org.springframework.web.bind.annotation.RequestPart;

**import** org.springframework.web.multipart.MultipartFile;

**import** org.springframework.web.servlet.ModelAndView;

**import** com.doj.spring.web.bean.Student;

@Controller

**public** **class** WebController {

//multiple mapping with one remote method

@RequestMapping(value={"/", "/index","/home","/welcome"})

**public** String home(){

**return** "home";

}

//With Model and Model Name to View Resolver

@RequestMapping("/indexc")

**public** ModelAndView welcome(){

Map<String, String> model = **new** HashMap<>();

model.put("name", "Sumit");

**return** **new** ModelAndView("home","model", model);

}

//We are using Spring ModelMap for return the model value

@RequestMapping("/hello")

**public** String index(ModelMap model){

model.put("name", "Sumit");

**return** "home";

}

//We are using Spring ModelMap and fetching request parameter here

@RequestMapping("/doj")

**public** String hello(ModelMap model, HttpServletRequest request){

String name = request.getParameter("name");

model.put("name", name);

**return** "home";

}

//We are using Spring ModelMap and Mapping the attribte with request param annotation with attributes whatever is ur requirement

@RequestMapping("/dojc")

**public** String doj(ModelMap model, @RequestParam(defaultValue = "DOJ Students", required=**true**, value="fname") String name,

@RequestParam(required = **false**, value="lname") String sname){

**if**(sname != **null**){

name = name +" "+ sname;

}

model.put("name", name);

**return** "home";

}

@RequestMapping("/doj-student-{fname}-{lname}")

**public** String dojStudent(ModelMap model, @PathVariable(value="fname") String name, @PathVariable(value="lname") String sname,

@RequestParam String address){

**if**(sname != **null**){

name = name +" "+ sname;

}

model.put("name", name);

**return** "home";

}

@RequestMapping(value="/doj-student", method=RequestMethod.GET)

**public** String getDojStudent(ModelMap model, Student student){

String name = **null**;

**if**(student.getFname() != **null**){

name = student.getFname();

}

**if**(student.getLname() != **null**){

name = name + " " +student.getLname();

}

**if**(student.getAddress() != **null**){

name = name+" "+student.getAddress();

}

**if**(student.getCourse() != **null**){

name = name+" "+student.getCourse();

}

model.put("name", name);

**return** "home";

}

@RequestMapping(value = "/dojstudent", method=RequestMethod.GET)

**public** String student(){

**return** "student";

}

@RequestMapping(value="/dojstudent", method=RequestMethod.POST)

**public** String getStudent(ModelMap model, @Valid Student student, Errors errors, @RequestPart("profilePicture") MultipartFile profilePicture){

**if**(errors.hasErrors()){

System.out.println("Errors "+errors.getErrorCount());

**return** "student";

}

**try** {

profilePicture.transferTo(**new** File("E:\\Shiksha\\Programming\\DOJWorkspace\\"+profilePicture.getOriginalFilename()));

} **catch** (IllegalStateException | IOException e) {

e.printStackTrace();

}

String name = **null**;

**if**(student.getFname() != **null**){

name = student.getFname();

}

**if**(student.getLname() != **null**){

name = name + " " +student.getLname();

}

**if**(student.getAddress() != **null**){

name = name+" "+student.getAddress();

}

**if**(student.getCourse() != **null**){

name = name+" "+student.getCourse();

}

model.put("name", name);

**return** "home";

}

}

Footer.jsp

<h1>&copy; DOJ SOFTWARE CONSULTANTS</h1>

Header.jsp

<h1>DOJ STUDENT FORM</h1>

mainTempalte.jsp

<!DOCTYPE h1 PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<%@taglib uri=*"http://tiles.apache.org/tags-tiles"* prefix=*"tiles"* %>

<html>

<head>

<title><tiles:insertAttribute name=*"title"*/></title>

<style>

*.error* {

color: *red*; font-weight: *bold*;

}

</style>

</head>

<body style="text-align: *center*;">

<div id=*"header"* style="height: *10%*;background-color: *gray*;"><tiles:insertAttribute name=*"header"*/></div>

<div id=*"body"* style="height: *70%*; background-color: *aqua*;padding-top: *40px*;"><tiles:insertAttribute name=*"body"*/></div>

<div id=*"footer"* style="height: *10%*;background-color: *gray*;"><tiles:insertAttribute name=*"footer"*/></div>

</body>

</html>

Tiles-def.jsp

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE tiles-definitions PUBLIC

"-//Apache Software Foundation//DTD Tiles Configuration 2.0//EN"

"http://tiles.apache.org/dtds/tiles-config\_2\_0.dtd">

<tiles-definitions>

<definition name=*"mainTemplate"* template=*"/WEB-INF/view/templates/mainTemplate.jsp"*>

<put-attribute name=*"title"* value=*""*></put-attribute>

<put-attribute name=*"header"* value=*"/WEB-INF/view/templates/header.jsp"*></put-attribute>

<put-attribute name=*"body"* value=*""*></put-attribute>

<put-attribute name=*"footer"* value=*"/WEB-INF/view/templates/footer.jsp"*></put-attribute>

</definition>

<definition name=*"student"* extends=*"mainTemplate"*>

<put-attribute name=*"title"* value=*"STUDENT FORM"*></put-attribute>

<put-attribute name=*"body"* value=*"/WEB-INF/view/student.jsp"*></put-attribute>

</definition>

<definition name=*"home"* extends=*"mainTemplate"*>

<put-attribute name=*"body"* value=*"/WEB-INF/view/home.jsp"*></put-attribute>

<put-attribute name=*"title"* value=*"WELCOME DOJ CLASS"*></put-attribute>

</definition>

</tiles-definitions>

Home.jsp

*<h1>${name} Welcome to DOJ Classes for Spring MVC!!!</h1>*

*Student.jsp*

<%@ taglib prefix=*"form"* uri=*"http://www.springframework.org/tags/form"*%>

<form:form action=*"dojstudent"* commandName=*"student"* enctype=*"multipart/form-data"*>

<table style="text-align: *center*; position: *absolute*;padding-left: *500px*;">

<tr>

<td>First Name : </td>

<td><input name=*"fname"* type=*"text"*></td>

<td><form:errors path=*"fname"* cssClass=*"error"*/></td>

</tr>

<tr>

<td>Last Name : </td>

<td><input name=*"lname"* type=*"text"*></td>

<td><form:errors path=*"lname"* cssClass=*"error"*/></td>

</tr>

<tr>

<td>Address : </td>

<td><input name=*"address"* type=*"text"*></td>

<td><form:errors path=*"address"* cssClass=*"error"*/></td>

</tr>

<tr>

<td>Course : </td>

<td><input name=*"course"* type=*"text"*></td>

<td><form:errors path=*"course"* cssClass=*"error"*/></td>

</tr>

<tr>

<td>Age : </td>

<td><input name=*"age"* type=*"text"*></td>

<td><form:errors path=*"age"* cssClass=*"error"*/></td>

</tr>

<tr>

<td>Profile Picture : </td>

<td><input type=*"file"* name=*"profilePicture"* accept=*"image/jpeg,image/png,image/gif"* /></td>

<td></td>

</tr>

<tr>

<td></td>

<td><input type=*"Submit"* name=*"Submit"* value=*"Submit"*></td>

</tr>

</table>

</form:form>

Pom.xml

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>com.DOJSpring</groupId>

<artifactId>SpringWebCustomisigDispatcherServlet</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<name>SpringWebCustomisigDispatcherServlet</name>

<description>This application has configuration of dispatcher servlet with customization</description>

<properties>

<spring.version>4.0.0.RELEASE</spring.version>

<aspect.version>1.6.11</aspect.version>

<tiles.version>3.0.5</tiles.version>

<jstl.version>1.2</jstl.version>

</properties>

<dependencies>

<!-- Spring Core dependency -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-beans</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- Spring Core dependency -->

<!-- Spring AOP dependency -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aop</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjrt</artifactId>

<version>${aspect.version}</version>

</dependency>

<dependency>

<groupId>org.aspectj</groupId>

<artifactId>aspectjweaver</artifactId>

<version>${aspect.version}</version>

</dependency>

<!-- Spring AOP dependency -->

<!-- Spring MVC dependency -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- Spring MVC dependency -->

<!-- Tiles dependency -->

<dependency>

<groupId>org.apache.tiles</groupId>

<artifactId>tiles-core</artifactId>

<version>${tiles.version}</version>

</dependency>

<dependency>

<groupId>org.apache.tiles</groupId>

<artifactId>tiles-jsp</artifactId>

<version>${tiles.version}</version>

</dependency>

<dependency>

<groupId>org.apache.tiles</groupId>

<artifactId>tiles-api</artifactId>

<version>${tiles.version}</version>

</dependency>

<!-- Tiles dependency -->

<!-- JSTL dependency -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>${jstl.version}</version>

</dependency>

<!-- JSTL dependency -->

<dependency>

<groupId>javax.validation</groupId>

<artifactId>validation-api</artifactId>

<version>1.0.0.GA</version>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-validator</artifactId>

<version>4.2.0.Final</version>

</dependency>

</dependencies>

<build>

<sourceDirectory>src</sourceDirectory>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.1</version>

<configuration>

<source>1.7</source>

<target>1.7</target>

</configuration>

</plugin>

<plugin>

<artifactId>maven-war-plugin</artifactId>

<version>2.3</version>

<configuration>

<warSourceDirectory>WebContent</warSourceDirectory>

<failOnMissingWebXml>false</failOnMissingWebXml>

</configuration>

</plugin>

</plugins>

</build>

</project>