1. Tạo project bằng maven

* Group Id: chứa tên miền và domain.
* Artifact Id: tên dự án.
* Version: quan trọng sau khi bảo trì sẽ có các version khác nhau của dự án
* Packing: quyết định khi build thì file sẽ ra là đuôi gì

+ war: deploy bằng tomcat có layout view,….

+ jar: thường dành cho các file thư viện, thường dùng cho dynamic web.

<packaging>war</packaging>

Lưu ý: trong file pom.xml phải thêm packing war để build ra file war cho dự án nếu không có.

1. File pom.xml dùng để quản lý các version và các dependencies (các thư viện được dùng trong dự án) của dự án

* Để ý phiên bản java trong phần mềm với phiên bản java trong dự án phải giống nhau.

<properties>  
 <servlet.api.version>3.1.0</servlet.api.version>  
 <springframework.version>4.3.13.RELEASE</springframework.version>  
 <jsp.api.version>2.3.1</jsp.api.version>  
 <jstl.version>1.2</jstl.version>  
 <maven.compiler.source>1.8</maven.compiler.source>  
 <maven.compiler.target>1.8</maven.compiler.target>  
</properties>

<build>  
 <plugins>  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 <configuration>  
 <source>1.8</source>  
 <target>1.8</target>  
 </configuration>  
 </plugin>  
 </plugins>  
</build>

Code config dự án sao cho 2 version của máy và của dự án là giống nhau

* Library có 2 đối tượng không thể thiêu là servlet api và jsp api.

+ jsp chỉ thoả mãn khi view sử dụng jsp.(ngoài ra có nhiều view khác ví dụ thymleaf)

+ jsp không khác gì html thuần nhưng có thể sử dụng thêm 1 vài library riêng support cho jsp như jstl sẽ hỗ trợ xử lý logic.

<jsp.api.version>2.3.1</jsp.api.version>  
<jstl.version>1.2</jstl.version>

* Dependencies:

<dependencies>  
 <!-- Spring -->  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-core</artifactId>  
 <version>${springframework.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-web</artifactId>  
 <version>${springframework.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-webmvc</artifactId>  
 <version>${springframework.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-tx</artifactId>  
 <version>${springframework.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-orm</artifactId>  
 <version>${springframework.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context</artifactId>  
 <version>${springframework.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework</groupId>  
 <artifactId>spring-context-support</artifactId>  
 <version>${springframework.version}</version>  
 </dependency>  
  
 <!-- servlet api -->  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>javax.servlet-api</artifactId>  
 <version>${servlet.api.version}</version>  
 </dependency>  
  
 <!-- jsp api -->  
 <dependency>  
 <groupId>javax.servlet.jsp</groupId>  
 <artifactId>javax.servlet.jsp-api</artifactId>  
 <version>${jsp.api.version}</version>  
 </dependency>  
  
 <!-- jstl -->  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>jstl</artifactId>  
 <version>${jstl.version}</version>  
 </dependency>  
</dependencies>

Lưu ý: <verison>${tên biến}</version>.

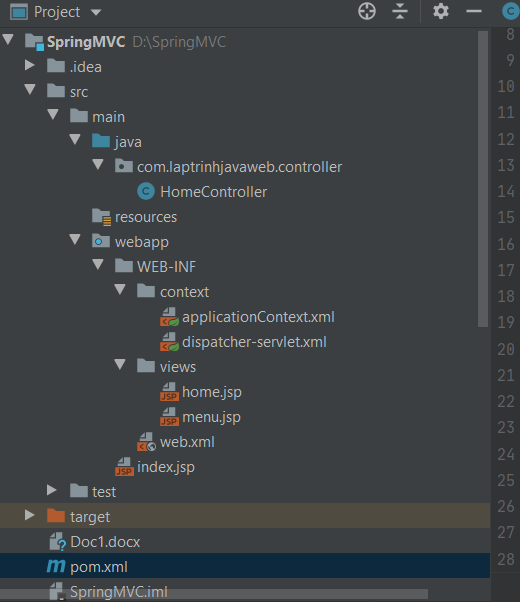
VD:<version>${jstl.version}</version>  
thường các version sẽ là <version>${1.3.2}</version> nhưng khi có quá nhiều dependency có phiên bản giống nhau ta sẽ khai báo số phiên bản giống như 1 biến   
tại properties rồi sau đó gán biến đó vào version như sau:

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

<!-- jstl -->  
 <dependency>  
 <groupId>javax.servlet</groupId>  
 <artifactId>jstl</artifactId>  
 <version>${jstl.version}</version>  
 </dependency>  
</dependencies>

1. Tạo folder webapp

* Tạo các file giống như bên dưới



* Trong folder webapp tạo 1 file index.jsp

<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>  
<**c:redirect** url="/trang-chu"/>

* Tạo folder context trong folder web-inf
* Trong folder context tạo 2 file applicationContext.xml và dispatcher-servlet.xml

applicationContext.xml

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  
 xmlns:context="http://www.springframework.org/schema/context"  
  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
  
 http://www.springframework.org/schema/beans/spring-beans-3.0.xsd  
  
 http://www.springframework.org/schema/context  
  
 http://www.springframework.org/schema/context/spring-context.xsd">  
  
 <!-- Activates scanning of @Repository and @Service and @Component -->  
  
 <context:component-scan base-package="com.laptrinhjavaweb" />  
  
</beans>

dispatcher-servlet.xml

<beans xmlns="http://www.springframework.org/schema/beans"  
  
 xmlns:context="http://www.springframework.org/schema/context"  
  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  
 xmlns:mvc="http://www.springframework.org/schema/mvc"  
  
 xsi:schemaLocation="  
  
 http://www.springframework.org/schema/beans  
  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
  
 http://www.springframework.org/schema/mvc  
  
 http://www.springframework.org/schema/mvc/spring-mvc.xsd  
  
 http://www.springframework.org/schema/context  
  
 http://www.springframework.org/schema/context/spring-context.xsd">  
  
 <context:component-scan  
 base-package="com.laptrinhjavaweb.controller" />  
  
 <mvc:annotation-driven />  
  
 <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  
  
 <property name="prefix" value="/WEB-INF/views/" />  
  
 <property name="suffix" value=".jsp" />  
  
 </bean>  
</beans>

* Trong file views tạo file home.jsp

<%--  
 Created by IntelliJ IDEA.  
 User: trant  
 Date: 7/7/2021  
 Time: 10:19 AM  
 To change this template use File | Settings | File Templates.  
--%>  
<%@ **page** contentType="text/html;charset=UTF-8" language="java" pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
  
<html>  
  
<head>  
  
 <meta charset="UTF-8">  
  
 <title>Home Page</title>  
  
</head>  
  
<body>  
<h1>Trang chủ</h1>  
<h1>Menu</h1>  
  
<%@ **include** file="menu.jsp" %>  
</body>  
  
</html>

* Tạo file index.jsp

<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>  
<**c:redirect** url="/trang-chu"/>

* Trong thư mục src/main/java tạo package là com.laptrinhjavaweb.controller tạo file HomeController.java.

(Mô hình MVC: Model là dữ liệu database đứng ở giữa tương tác với controller và view. Khi user sử dụng view tạo 1 request tử url về phía server sẽ nhận và trả về kết quả. Khi server dò kết quả sẽ tìm ở @Controller trước nếu thấy sẽ dò đến url cần tìm)

package com.laptrinhjavaweb.controller;  
  
  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestMethod;  
import org.springframework.web.servlet.ModelAndView;  
//Để file java thông thường trở thành 1 controller ta thêm @Controller  
   
@Controller  
public class HomeController {  
  
 @RequestMapping(value = "/trang-chu", method = RequestMethod.*GET*)  
 public ModelAndView homePage(){  
 ModelAndView mav = new ModelAndView("home");  
 return mav;  
 }  
  
}

@RequestMapping: là nơi nhận các url

Value: là 1 phần của url định hướng địa chỉ cần đến của request

Method: bao gồm các dạng chủ yếu get, post, request,… Các phương thức này sẽ giao tiếp với database để sử dụng dữ liệu

ModelAndView: cho phép controller trả về giá trị bao gồm cả model và view.

Sau khi tìm được request chính xác ModelAndView sẽ tìm đến file .jsp cần thiết để thực hiện trả về giao diện. Ở dòng code ModelAndView mav = new ModelAndView("home"); đường dẫn chính xác phải là /webapp/WEB-INF/views/home.jsp nhưng chúng ta có thể config lại đường dẫn này ngắn gọn hơn trong dispatcher-servlet.xml.

<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  
  
 <property name="prefix" value="/WEB-INF/views/" />  
  
 <property name="suffix" value=".jsp" />  
  
</bean>

Trong dòng code trên ta thấy prefix là để config đường dẫn file chứa view còn suffix là để config đuôi của những file được gọi đến.

Chương trình có thể hiểu đc đường dẫn này là do nó đã được định nghĩa thông qua đối tượng InternalResourceViewResolver.

1. Chi tiết các file:

* Trong file web.xml có đoạn code sau

<welcome-file-list>  
 <welcome-file>index.jsp</welcome-file>  
</welcome-file-list>

Đoạn code này sẽ chỉ đường dẫn cho file đầu tiên mà chương trình truy cập ở đây là index.jsp và file này sẽ chuyển đường dẫn đến trang chủ bằng đoạn code sau (trong file index.jsp)

<%@ **taglib** prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>  
<**c:redirect** url="/trang-chu"/>

Và sau khi redirect về trang chủ thì @RequestMapping từ file HomeController.java sẽ gọi tới file home.jsp để hiện view lên màn hình.

* Để chương trình biết được controller nằm ở đâu chúng ta cần config tại dispatcher servlet.xml với đoạn code sau:

<context:component-scan  
 base-package="com.laptrinhjavaweb.controller" />

<mvc:annotation-driven />

Trong dòng <mvc: annotation-driven/> có tác dụng định nghĩa các annotation như @Controller và @Requestmapping để sử dụng trong springmvc.

* 2 file dispatcher-servlet.xml và applicationContext.xml ban đâu mới chỉ ở dạng khai báo muốn sử dụng nó ta cần phải config tại file web.xml.

<listener>  
 <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  
</listener>

Khi ta tạo 1 request đến chương trình tức là hệ thống phải listen đc url đó dòng code trên sẽ giúp chúng ta thực hiện được điều đó.

<context-param>  
 <param-name>contextConfigLocation</param-name>  
 <param-value>  
 /WEB-INF/context/applicationContext.xml  
 </param-value>  
</context-param>

Dòng Code trên dùng để load file applicationContext.xml.

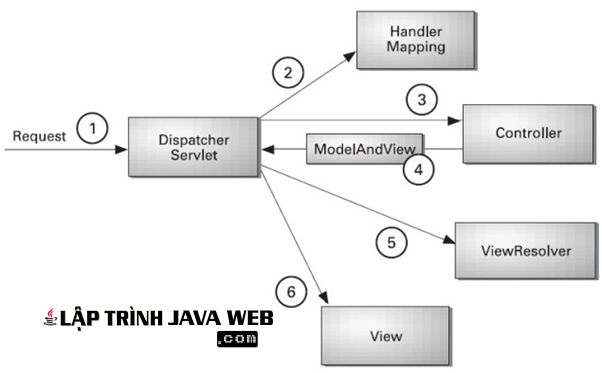
<servlet>  
 <servlet-name>dispatcher</servlet-name>  
 <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  
 <init-param>  
 <param-name>contextConfigLocation</param-name>  
 <param-value>/WEB-INF/context/dispatcher-servlet.xml</param-value>  
 </init-param>  
 <load-on-startup>1</load-on-startup>  
</servlet>  
<servlet-mapping>  
 <servlet-name>dispatcher</servlet-name>  
 <url-pattern>/</url-pattern>  
</servlet-mapping>

Dòng code trên dùng để load file dispatcher-servlet.xml.

<session-config>  
 <tracking-mode>COOKIE</tracking-mode>  
</session-config>

Dòng code trên để remove jsonid ở đuôi url.

1. Spring MVC flow:



Luồng chạy của spring MVC:

Khi 1 request gửi tới thì sẽ vào đối tượng đầu tiên là dispatcher Servlet và từ đó hệ thống sẽ đi tìm những handler mapping (là những request mapping) ở controller. Sau khi tìm được request mapping thích hợp hệ thống trả về ModelAndView được định nghĩa trong request mapping đó. Ngoài ra, khi gọi đến ModelAndView thì nó sẽ trả về ViewResolver (là đối tượng chịu trách nhiệm lấy các đối tượng View cho 1 đối tượng và vị trí cụ thể). Cuối cùng hệ thống sẽ trả về mặt view đã tìm kiếm.

File applicationContext.xml được định nghĩa là 1 bean container.

1. Load menu tự động sử dụng interceptor