[Bài đọc] Ứng dụng Spring MVC đầu tiên

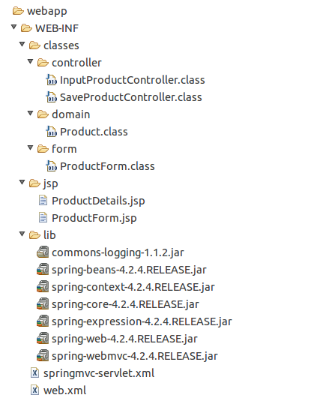
*Nguồn:*[*Spring MVC: A Tutorial (Second Edition)*](https://books.google.com.vn/books?id=O1LzCwAAQBAJ&printsec=frontcover#v=onepage&q&f=false)*, Paul Deck*

Ứng dụng **springmvc-intro1** giới thiệu một ứng dụng Spring MVC cơ bản. Nó cho bạn thấy cách hoạt động của Spring MVC.

**Cấu trúc thư mục**

Hình 3.1 cho thấy cấu trúc thư mục của **springmvc-intro1**. Lưu ý rằng thư mục WEB-INF/lib chứa tất cả các tệp jar được yêu cầu bởi Spring MVC, trong đó tệp spring-webmvc-x.y.z.jar, chứa lớp DispatcherServlet. Cũng lưu ý rằng Spring MVC phụ thuộc vào thành phần Apache Commons Logging và nếu không có nó, các ứng dụng Spring MVC của bạn sẽ không hoạt động. Bạn có thể tải xuống thành phần này từ trang sau:

<http://commons.apache.org/proper/commons-logging/>



Hình 3.1: Cấu trúc thư mục của springmvc-intro1

Các trang JSP cho ứng dụng này được lưu trữ trong /WEB-INF/jsp.

**Trình mô tả triển khai và tệp cấu hình Spring MVC**

Bây giờ hãy xem bộ mô tả triển khai (tệp web.xml) trong VD 3.1

VD 3.1: Bộ mô tả triển khai cho springmvc-intro1

<?xml version="1.0" encoding="UTF-8"?>  
<web-app version="3.1"  
 xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee   
 http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">  
  
<servlet>  
<servlet-name>springmvc</servlet-name>  
<servlet-class  
org.springframework.web.servlet.DispatcherServlet  
</servlet-class>  
<load-on-startup>1</load-on-startup>  
</servlet>  
  
<servlet-mapping>  
<servlet-name>springmvc</servlet-name>  
<!-- map all requests to the DispatcherServlet -->  
<url-pattern>/</url-pattern>  
</servlet-mapping>  
  
</web-app>

Bạn đang nói với servlet/JSP rằng, bạn muốn sử dụng trình điều phối Spring MVC và ánh xạ nó tới tất cả các URL bằng cách sử dụng kí hiệu "/" trong phần tử < url-pattern>. Vì không có phần tử <init-param> trong phần tử <servlet>, tệp cấu hình Spring MVC được giả định là nằm trong /WEB-INF và tuân theo quy ước đặt tên thông thường.

Tiếp theo, kiểm tra tệp cấu hình Spring MVC (tệp springmvc-servlet.xml) trong VD 3.2.

VD 3.2: Tệp cấu hình Spring MVC

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd">  
  
<bean name="/input-product"  
class="controller.InputProductController"/>  
<bean name="/save-product"  
class="controller.SaveProductController"/>  
  
</beans>

Ở đây, bạn khai báo hai controller, InputProductController và SaveProductController, và ánh xạ chúng vào /input-product và /save-product, tương ứng.

**Controllers**

Ứng dụng springmvc-intro1 có hai controller “kiểu cũ”, InputProductController và SaveProductController. Cả hai đều thi hành interface controller. Lớp InputProductController được đưa ra trong VD 3.3 và SaveProductControler trong VD 3.4.

VD 3.3: Lớp InputProductController

package controller;  
  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import org.apache.commons.logging.Log;  
import org.apache.commons.logging.LogFactory;  
import org.springframework.web.servlet.ModelAndView;  
import org.springframework.web.servlet.mvc.Controller;  
  
public class InputProductController implements Controller {  
  
 private static final Log *logger* = LogFactory  
 .getLog(InputProductController.class);  
  
 @Override  
 public ModelAndView handleRequest(HttpServletRequest request,  
 HttpServletResponse response) throws Exception {  
 *logger*.info("InputProductController called");  
 return new ModelAndView("/WEB-INF/jsp/ProductForm.jsp");  
 }  
}

Phương thức handleRequest của lớp InputProductController trả về một ModelAndView có chứa một view không có model. Trong trường hợp này, yêu cầu sẽ được chuyển tiếp đến trang /WEB-INF/jsp/ProductForm.jsp.

VD 3.4.1: Lớp ProductForm

package controller;  
  
public class ProductForm {  
 private String name;  
 private String description;  
 private String price;  
  
 public ProductForm() {  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getDescription() {  
 return description;  
 }  
  
 public void setDescription(String description) {  
 this.description = description;  
 }  
  
 public String getPrice() {  
 return price;  
 }  
  
 public void setPrice(String price) {  
 this.price = price;  
 }  
  
}

VD 3.4.2: Lớp SaveProductController

package controller;  
  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import org.apache.commons.logging.Log;  
import org.apache.commons.logging.LogFactory;  
import org.springframework.web.servlet.ModelAndView;  
import org.springframework.web.servlet.mvc.Controller;  
import domain.Product;  
import form.ProductForm;  
  
public class SaveProductController implements Controller {  
  
 private static final Log *logger* = LogFactory  
 .getLog(SaveProductController.class);  
  
 @Override  
 public ModelAndView handleRequest(HttpServletRequest request,  
 HttpServletResponse response) throws Exception {  
 *logger*.info("SaveProductController called");  
 ProductForm productForm = new ProductForm();  
 // populate action properties  
 productForm.setName(request.getParameter("name"));  
 productForm.setDescription(request.getParameter(  
 "description"));  
 productForm.setPrice(request.getParameter("price"));  
  
 // create model  
 Product product = new Product();  
 product.setName(productForm.getName());  
 product.setDescription(productForm.getDescription());  
 try {  
 product.setPrice(  
 Float.*parseFloat*(productForm.getPrice()));  
 } catch (NumberFormatException e) {  
 }  
  
 // insert code to save Product  
  
 return new ModelAndView("/WEB-INF/jsp/ProductDetails.jsp",  
 "product", product);  
 }  
}

Phương thức handleRequest trong lớp SaveProductController tạo ra một đối tượng ProductForm và thiết lập giá trị cho các thuộc tính của nó bằng các tham số request. Sau đó, nó tạo ra một đối tượng Product nhận các giá trị thuộc tính của nó từ ProductForm. Vì thuộc tính price của ProductForm là một String và ánh xạ của nó trong lớp Product có kiểu float nên chúng ta sử dụng hàm Float.parseFloat() để ép kiểu.

Phương thức handleRequest trong SaveProductController kết thúc bằng cách trả về một ModelAndView có chứa một đường dẫn view, một tên model và một model (đối tượng Product). Model được thêm vào đối tượng ModelAndView sẽ có sẵn cho view để hiển thị.

**View**

Ứng dụng springmvc-intro1 đi kèm với hai trang JSP, trang ProductForm.jsp (VD 3.5) và trang ProductDetails.jsp (VD 3.6).

VD 3.5: Trang ProductForm.jsp

<!DOCTYPE html>  
<html>  
<head>  
 <title>Add Product Form</title>  
 <style type="text/css">@import url(css/main.css);</style>  
</head>  
<body>  
  
<div id="global">  
 <form action="save-product" method="post">  
 <fieldset>  
 <legend>Add a product</legend>  
 <label for="name">Product Name: </label>  
 <input type="text" id="name" name="name" value=""  
 tabindex="1">  
 <label for="description">Description: </label>  
 <input type="text" id="description" name="description"  
 tabindex="2">  
 <label for="price">Price: </label>  
 <input type="text" id="price" name="price" tabindex="3">  
 <div id="buttons">  
 <label for="dummy"> </label>  
 <input id="reset" type="reset" tabindex="4">  
 <input id="submit" type="submit" tabindex="5"  
 value="Add Product">  
 </div>  
 </fieldset>  
 </form>  
</div>  
</body>  
</html>

VD 3.6: Trang ProductDetails.jsp

<!DOCTYPE html>  
<html>  
<head>  
 <title>Save Product</title>  
 <style type="text/css">@import url(css/main.css);</style>  
</head>  
<body>  
<div id="global">  
 <h4>The product has been saved.</h4>  
 <p>  
 <h5>Details:</h5>  
 Product Name: ${product.name}<br/>  
 Description: ${product.description}<br/>  
 Price: $${product.price}  
 </p>  
</div>  
</body>  
</html>

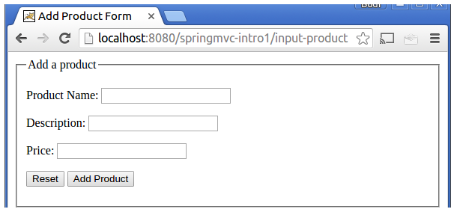
Trang ProductDetails.jsp có quyền truy cập vào đối tượng Product được truyền bởi SaveProductController với tên model là product. Chúng ta sử dụng các biểu thức của JSP Expression Language để hiển thị các thuộc tính khác nhau của đối tượng Product.

**Testing**

Để kiểm tra ứng dụng, hãy hướng trình duyệt của bạn đến URL này:

<http://localhost:8080/springmvc-intro1/input-product>

Bạn sẽ thấy form Product như trong Hình 3.2. Nhập giá trị vào các trường trống và nhấp vào nút Add Product. Bạn sẽ thấy các thuộc tính của sản phẩm được hiển thị trong trang tiếp theo.



Hình 3.2: Form Product trong springmvc-intro1

**Trình phân giải View**

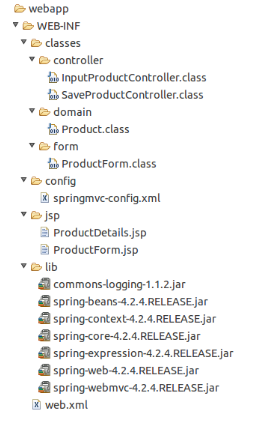
Trình phân giải view trong Spring MVC chịu trách nhiệm phân giải các view. Để sử dụng và cấu hình trình phân giải view, hãy khai báo một bean viewResolver trong tệp cấu hình của bạn.

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

Bean “viewResolver” ở trên cấu hình hai thuộc tính, tiền tố và hậu tố. Kết quả là các đường dẫn view của bạn sẽ ngắn hơn. Thay vì đặt đường dẫn view thành /WEB-INF/jsp/myPage.jsp, ví dụ, bạn chỉ cần viết myPage và trình phân giải view sẽ thêm tiền tố và hậu tố.

Ví dụ, hãy xem xét ứng dụng springmvc-intro2, tương tự như springmvc-intro1. Tuy nhiên, tên và vị trí của tập tin cấu hình đã được thay đổi. Ngoài ra, nó sẽ cấu hình trình phân giải view mặc định để thêm tiền tố và hậu tố cho tất cả các đường dẫn view.

Hình 3.3 cho thấy cấu trúc thư mục của springmvc-intro2.



Hình 3.3: Cấu trúc thư mục của springmvc-intro2

Trong tệp tin springmvc-intro2, tệp cấu hình Spring MVC đã được đổi tên thành springmvc-config.xml và được chuyển đến / WEB-INF / config. Để cho Spring MVC biết nơi tìm nó, bạn cần chuyển vị trí của tệp tới trình điều phối Spring MVC. Ví dụ 3.7 cho thấy bộ mô tả triển khai (tệp tin web.xml) cho springmvc-intro2.

Ví dụ 3.7: Bộ mô tả triển khai cho springmvc-intro2

<?xml version="1.0" encoding="UTF-8"?>  
<web-app version="3.1"  
 xmlns="http://xmlns.jcp.org/xml/ns/javaee"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee  
http://xmlns.jcp.org/xml/ns/javaee/web-app\_3\_1.xsd">  
  
 <servlet>  
 <servlet-name>springmvc</servlet-name>  
 <servlet-class>  
 org.springframework.web.servlet.DispatcherServlet  
 </servlet-class>  
 <init-param>  
 <param-name>contextConfigLocation</param-name>  
 <param-value>  
 /WEB-INF/config/springmvc-config.xml  
 </param-value>  
 </init-param>  
 <load-on-startup>1</load-on-startup>  
 </servlet>  
  
 <servlet-mapping>  
 <servlet-name>springmvc</servlet-name>  
 <url-pattern>/</url-pattern>  
 </servlet-mapping>  
</web-app>

Chú ý đến phần tử <init-param> trong tệp web.xml. Để tham chiếu đến tệp cấu hình không sử dụng tên và vị trí mặc định, bạn cần sử dụng tham số ban đầu contextConfigLocation. Giá trị của nó phải là đường dẫn đến tệp cấu hình, liên quan đến thư mục ứng dụng.

Tệp cấu hình cho springmvc-intro2 được đưa ra trong VD 3.8.

VD 3.8: Tệp cấu hình Spring MVC cho springmvc-intro2

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd">  
  
 <bean name="/input-product"  
 class="controller.InputProductController"/>  
 <bean name="/save-product"  
 class="controller.SaveProductController"/>  
 <bean id="viewResolver"  
 class="org.springframework.web.servlet.view.  
InternalResourceViewResolver">  
 <property name="prefix" value="/WEB-INF/jsp/"/>  
 <property name="suffix" value=".jsp"/>  
 </bean>  
</beans>

Để kiểm tra ứng dụng, hãy hướng trình duyệt của bạn đến URL này:

<http://localhost:8080/springmvc-intro2/input-product>

Bạn sẽ thấy một form tương tự như Hình 3.2.