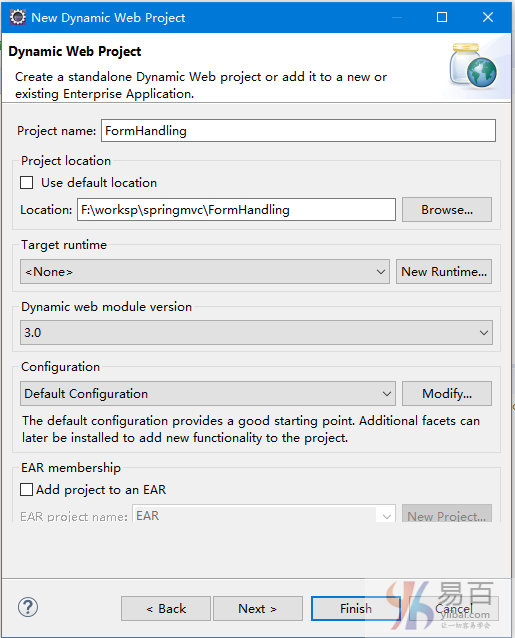
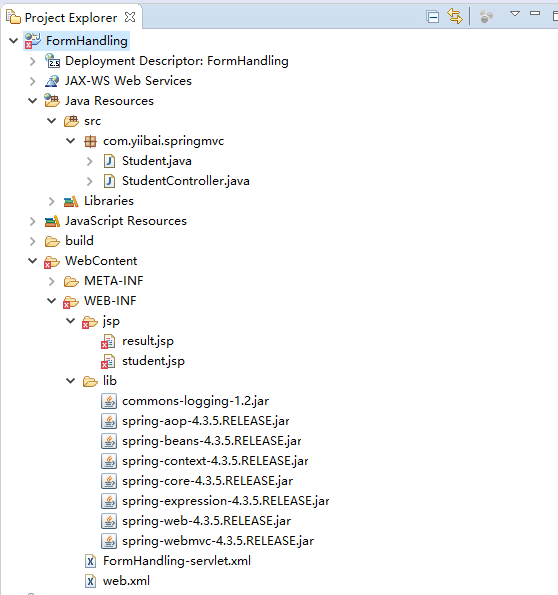
以下示例演示如何编写一个简单的基于Web的应用程序，它使用**Spring Web MVC**框架使用HTML表单。 首先使用Eclipse IDE，并按照以下步骤使用**Spring Web Framework**开发基于动态表单的**Web**应用程序：

1. 基于上一小节中的**Spring MVC - Hello World实例**章节所创建的 **HelloWeb**来创建一个新的工程为：FormHandling，并创建一个包名称为 com.yiibai.springmvc。  
   
2. 在com.yiibai.springmvc包下创建两个Java类Student，StudentController。
3. 在jsp子文件夹下创建两个视图文件student.jsp，result.jsp。
4. 最后一步是创建所有源和配置文件的内容并运行应用程序，如下所述。

完整的项目文件结构如下所示 -



**Student.java**文件中的代码内容 -

package com.yiibai.springmvc;

public class Student {

private Integer age;

private String name;

private Integer id;

public void setAge(Integer age) {

this.age = age;

}

public Integer getAge() {

return age;

}

public void setName(String name) {

this.name = name;

}

public String getName() {

return name;

}

public void setId(Integer id) {

this.id = id;

}

public Integer getId() {

return id;

}

}

Java

**StudentController.java** 文件中的代码内容 -

package com.yiibai.springmvc;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.ModelAttribute;

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

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

import org.springframework.web.servlet.ModelAndView;

import org.springframework.ui.ModelMap;

@Controller

public class StudentController {

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

public ModelAndView student() {

return new ModelAndView("student", "command", new Student());

}

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

public String addStudent(@ModelAttribute("SpringWeb")Student student,

ModelMap model) {

model.addAttribute("name", student.getName());

model.addAttribute("age", student.getAge());

model.addAttribute("id", student.getId());

return "result";

}

}

Java

这里的第一个服务方法student()，我们已经在ModelAndView对象中传递了一个名为“command”的空对象，因为如果在JSP中使用<form：form>标签，**spring**框架需要一个名为“command”的对象文件。 所以当调用student()方法时，它返回student.jsp视图。

第二个服务方法addStudent()将在 URLHelloWeb/addStudent上的POST方法提交时调用。将根据提交的信息准备模型对象。最后，将从服务方法返回“result”视图，这将最终渲染result.jsp视图。

**student.jsp**文件的内容如下所示 -

<%@ page contentType="text/html; charset=UTF-8" %>

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

<html>

<head>

<title>Spring MVC表单处理</title>

</head>

<body>

<h2>Student Information</h2>

<form:form method="POST" action="/FormHandling/addStudent">

<table>

<tr>

<td><form:label path="name">名字：</form:label></td>

<td><form:input path="name" /></td>

</tr>

<tr>

<td><form:label path="age">年龄：</form:label></td>

<td><form:input path="age" /></td>

</tr>

<tr>

<td><form:label path="id">编号：</form:label></td>

<td><form:input path="id" /></td>

</tr>

<tr>

<td colspan="2">

<input type="submit" value="提交表单"/>

</td>

</tr>

</table>

</form:form>

</body>

</html>

HTML

**result.jsp**文件的内容如下 -

<%@ page contentType="text/html; charset=UTF-8" %>

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

<html>

<head>

<title>Spring MVC表单处理</title>

</head>

<body>

<h2>提交的学生信息如下 - </h2>

<table>

<tr>

<td>名称：</td>

<td>${name}</td>

</tr>

<tr>

<td>年龄：</td>

<td>${age}</td>

</tr>

<tr>

<td>编号：</td>

<td>${id}</td>

</tr>

</table>

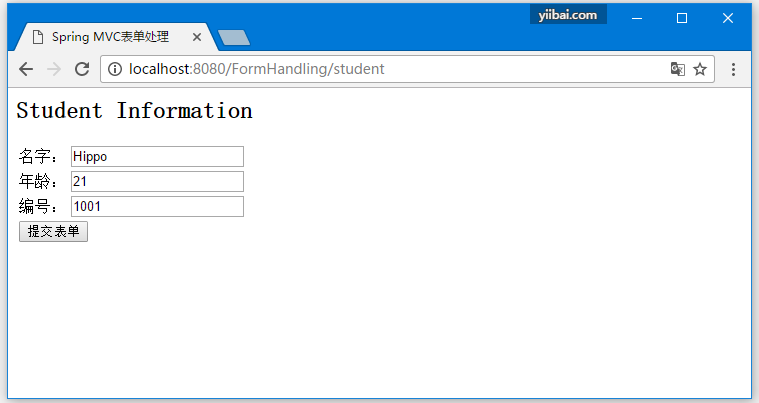
</body>

</html>

HTML

完成创建源和配置文件后，导出应用程序。 右键单击应用程序，并使用**导出> WAR文件**选项，并将 FormHandling.war 文件保存在Tomcat的webapps文件夹中。或者直接右键选择“**Run As -> Run On Server**”。

启动Tomcat服务器，并确保您能够使用标准浏览器从**webapps**文件夹访问其他网页。现在尝试URL => [http://localhost:8080/FormHandling/student](http://localhost:8080/FormHandling/student" \t "_blank) ，如果**Spring Web**应用程序没有问题，那么应该看到以下结果：



提交所需信息后，点击提交按钮提交表单。 如果**Spring Web**应用程序没有问题，应该看到以下结果：

