实验08 Spring MVC框架：基本配置及应用

**【实验目的及要求】**

1. 理解Spring MVC框架的基本原理；
2. 掌握Spring MVC的核心类及注解；
3. 要求所有在表格中填入的信息，均为5号字体；
4. 要求所有回答的文本格式：**五号，宋体、1.5倍行距，保留段单元格背景。**

**【实验步骤】**

**1、创建和运行Maven Web应用（用户登录）**

（1）新建Maven Web项目ex08，注意在“New Project”对话框中勾选“Create from archetype”复选框，并在列表中选择“org.apache.maven.archetypes: maven-archetype-webapp”选项，如图8.1所示。

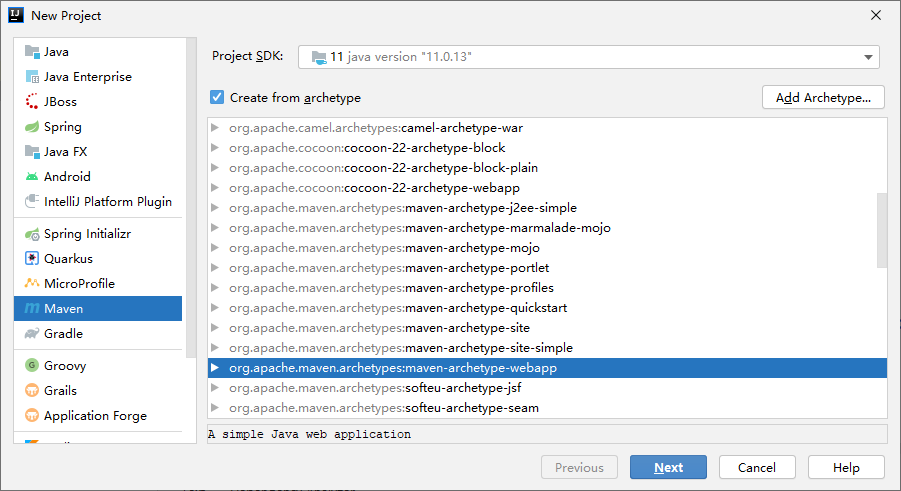


图8.1 “New Project”对话框

（2）在pom.xml文件的<dependencies/>元素中加入如下依赖信息，内容如下。

|  |
| --- |
| <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>5.2.8.RELEASE</version> </dependency> *<!--Spring MVC-->* <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-webmvc</artifactId>  <version>5.2.8.RELEASE</version> </dependency> *<!-- servlet -->* <dependency>  <groupId>javax.servlet</groupId>  <artifactId>javax.servlet-api</artifactId>  <version>3.1.0</version>  <scope>provided</scope> </dependency> *<!--JSP-->* <dependency>  <groupId>javax.servlet.jsp</groupId>  <artifactId>jsp-api</artifactId>  <version>2.1</version>  <scope>provided</scope> </dependency> |

（3）在pom.xml文件的<plugins/>元素中加入tomcat插件配置信息，内容如下。

|  |
| --- |
| <plugin>  <groupId>org.apache.tomcat.maven</groupId>  <artifactId>tomcat7-maven-plugin</artifactId>  <configuration>  <port>8080</port>  <path>/javaee\_ex08</path>  </configuration> </plugin> |

（4）在src/main/resources中创建config文件夹，并在其中创建Spring MVC配置文件spring-mvc.xml，内容如下。

|  |
| --- |
| *<?*xml version="1.0" encoding="UTF-8"*?>* <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"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  **http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd**"> <context:component-scan base-package="com.javaee.ex08.controller"/> <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property name="prefix" value="/WEB-INF/pages/"/>  <property name="suffix" value=".jsp"/>  </bean> </beans> |

**说明**：spring-mvc.xml中的配置信息主要有两项：一是注解的扫描配置，二是配置视图解析器Bean。因为用到了注解，故需要配置命名空间及Schema约束信息，详见粗体信息。若不采用注解，则要使用<bean/>元素配置每个控制器类。视图解析器是Spring框架提供的类，故无法使用注解。视图解析器中的前缀（prefix）和后缀（suffix）是Spring MVC查找视图的依据。

（5）在web.xml文件中，加入前端控制器的配置。

|  |
| --- |
|  |

（6）创建一个控制器com.javaee.ex08.controller.UserController类，用于账户管理、登录及注册（为简单起见，这里仅给出登录），内容如下。

|  |
| --- |
| package com.javaee.ex08.controller;  import org.springframework.stereotype.Controller; import org.springframework.ui.Model; import org.springframework.web.bind.annotation.RequestMapping; import org.springframework.web.bind.annotation.RequestParam;  @Controller @RequestMapping("/user") // 公共请求路径 public class UserController {  @RequestMapping("/to\_login")  public String toLogin(){  return "user/login"; // 经过视图解析器解析  }  @RequestMapping("/login")  public String login(@RequestParam("login\_name") String loginName,  String password, Model model){  if (!password.equals("888888")){  model.addAttribute("login\_error","登录失败，请重试！");  return "forward:to\_login"; *// 转发请求（不含公共路径）  //return "redirect:to\_login"; // 重定向请求（不含公共路径）*  }  model.addAttribute("login\_name",loginName);  model.addAttribute(password);  return "user/main";  }  @RequestMapping("/to\_index")  public String toIndex(){  return "redirect:/index.jsp"; // 重定向到页面，不被解析，“/”表示从根相对路径开始，若无“/”，则从当前公共路径“user”开始  } } |

**说明**：控制器方法的return语句中，若含redirect或forward，则不受视图解析器解析。

（7）修改index.jsp文件（该文件在创建项目时，自动生成），在其中添加一超链接，内容如下。

|  |
| --- |
| <html> <body> <h2>Hello World!</h2> <a href="user/to\_login">登录</a> </body> </html> |

（8）在WEB-INF/pages下，新建login.jsp和main.jsp文件，用作登录页面及成功登录后要跳转的页面，内容如下。

|  |
| --- |
| *<!-- login.jsp -->*  <%@ page contentType="text/html;charset=UTF-8" language="java" %> <%@ page isELIgnored ="false" %> <html> <head>  <title>用户登录</title> </head> <body>  <form id="form1" name="form1" method="post" action="login">  <table width="344" height="95" border="0" cellpadding="0" cellspacing="1"  style="background-color: #3A8ECD; color: #000000;">  <tr>  <td height="30" colspan="2" align="center" valign="middle" style="color: #FFF">用户登录</td>  </tr>  <tr>  <td width="114" height="30" align="right" valign="middle" bgcolor="#FFFFFF">登录名：</td>  <td width="227" height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input type="text" name="login\_name" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">密码：</td>  <td height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input type="password" name="password" /></td>  </tr>  <tr>  <td height="30" colspan="2" align="center" valign="middle" bgcolor="#FFFFFF">  <input type="submit" name="button" id="button" value="提交" />&nbsp;&nbsp;  <input type="reset" name="button2" id="button2" value="重置" /></td>  </tr>  </table>  <p style="color: #F00">${login\_error}</p>  </form> </body> </html> |
| *<!-- main.jsp -->*  <%@ page contentType="text/html;charset=UTF-8" language="java" %> <%@ page isELIgnored ="false" %>  <html> <head>  <title>成绩管理系统</title> </head> <body> <p>[${login\_name}]用户您好！欢迎使用成绩管理系统</p> <p><a href="${pageContext.request.contextPath}/user/to\_index">回到首页</a></p> </body> </html> |

（9）配置运行参数，选择“run→Edit Configurations”菜单项，弹出“Run/Debug Configurations”对话框，如图8.1所示。

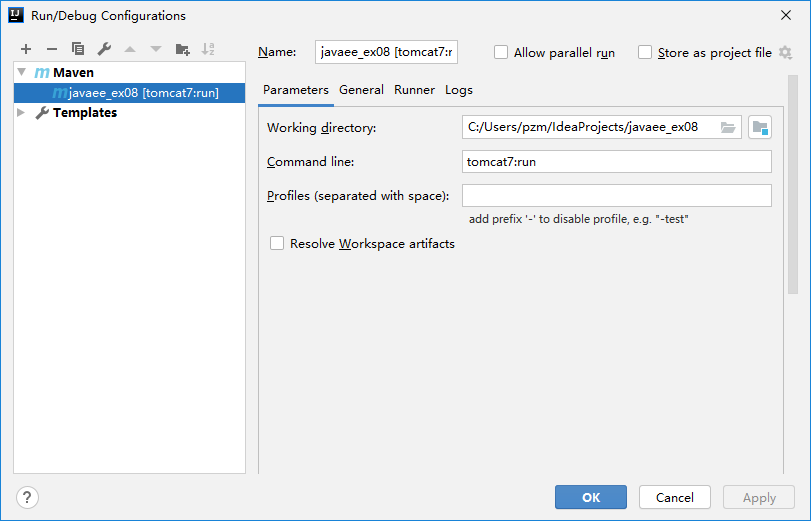


图8.1 “Run/Debug Configurations”对话框

（10）单击“+”按钮，在弹出的菜单中选择“Maven”菜单项，然后在“Command line”文本框中输入“tomcat7:run”，并确保“Working directory”文本框中为项目路径。单击“Ok”按钮，关闭对话框。

（11）运行项目，完成从index页到login页，从login页到login页（账号错误时），从login页到main页，以及从main页到index页的跳转。并给出每个页面的截图。

|  |
| --- |
| // index.jsp页面截图 |
| // login.jsp页面截图（正常） |
| // login.jsp页面截图（含错误提示信息） |
| // main.jsp页面截图 |

**2、实现用户注册交互**

在第1个实验的基础上，参考用户登录流程，实现用户注册过程的交互。注册信息包括用户名、登录名、密码（密码要有确认）、邮箱等信息。要求在注册（register.jsp）页面通过表单提交注册信息，由控制器方法接收注册数据，若注册信息中含有空字符串或两次输入密码不等，则返回到注册页面（可提供未成功提供的注册信息，便于进一步修改），并给出合适的提示信息（以红色显示），若注册信息有效，则自动跳转到登录页面。

（1）参照登录过程，在UserController中添加方法，实现注册过程的跳转。修改后的UserController类代码。

|  |
| --- |
|  |

（2）在WEB-INF/pages中，添加register.jsp页面，提供注册表单（表单的action值要与后面的控制器方法对应）。

|  |
| --- |
| // register.jsp页面 |

（3）在index.jsp页面中，添加跳转到register.jsp页面的<a/>元素，内容如下。

|  |
| --- |
|  |

（4）运行项目，完成从index页到register页，从register页到register页（账号错误时），以及从register页到login页的跳转。给出跳转过程中每个页面的截图。

|  |
| --- |
| // index.jsp页面截图 |
| // register.jsp页面截图（正常） |
| // register.jsp页面截图（含错误提示信息） |

**3、学生信息提交**

仿照第1~2实验的步骤，新建一个Maven Web项目student\_ms（pom.xml文件中的依赖配置同第1个实验），完成学生信息（包括学号、姓名、性别、年龄、籍贯及电话等）的提交并显示。

（1）在wepapp目录下，创建jsp文件add\_student.jsp，提供提交所有学生信息的表单（表单的action值要与后面的控制器方法对应），内容如下。

|  |
| --- |
|  |

（2）在/WEB-INF/pages目录下，创建jsp文件student\_info.jsp，要求显示所有学生信息，内容如下。

|  |
| --- |
|  |

（3）在创建com.javaee.ex08.controller.StudentController控制器类，实现学生信息的接收，并转发给studentinfo.jsp文件，内容如下。

|  |
| --- |
|  |

（4）在main/resources/config目录下创建springmvc-config.xml，在其中配置控制器扫描（扫描com.javaee.ex08.controller包）及视图解析器（前缀为/WEB-INF/pages/，后缀为.jsp），内容如下。

|  |
| --- |
|  |

（5）在web.xml文件中，配置Spring MVC前端控制器，内容如下。

|  |
| --- |
|  |

（6）运行项目，将提交前表单页面（含提交信息）及提交后显示提交信息页面截图。

|  |
| --- |
| // add\_student.jsp |
| // student\_info.jsp |