Servlet 打造一个登入注册程序

1.pom.xml 依赖

```
<dependencies>
      <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.11</version>
        <scope>test</scope>
      </dependency>
        <dependency>
              <groupId>dom4j</groupId>
              <artifactId>dom4j</artifactId>
              <version>1.6.1</version>
        </dependency>
      <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>servlet-api</artifactId>
        <version>2.5</version>
      </dependency>
        <dependency>
              <groupId>commons-beanutils</groupId>
              <artifactId>commons-beanutils</artifactId>
              <version>1.9.3</version>
        </dependency>
      <!--数据库连接-->
      <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>5.0.2.RELEASE</version>
      </dependency>
      <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.21</version>
        <scope>runtime</scope>
      </dependency>
  </dependencies>
```

2.sql 语句

```
CREATE   TABLE   `login`   (
```

```
    `id`  varchar(100)  NOT  NULL,
    `userName`  varchar(22)  NOT  NULL,
    `userPwd`  varchar(22)  NOT  NULL,
    `email`  varchar(22)  NOT  NULL,
    `birthday`  date  NOT  NULL
)  ENGINE=InnoDB  DEFAULT  CHARSET=utf8
```

3.User

```
public class User implements Serializable {

    private static final long serialVersionUID = -4313782718477229465L;

    // 用户 ID
    private String id;
    // 用户名
    private String userName;
    // 用户密码
    private String userPwd;
    // 用户邮箱
    private String email;
    // 用户生日
    private Date birthday;

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getUserPwd() {
        return userPwd;
    }
```

```
    public void setUserPwd(String userPwd) {
        this.userPwd = userPwd;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public Date getBirthday() {
        return birthday;
    }

    public void setBirthday(Date birthday) {
        this.birthday = birthday;
    }

    @Override
    public String toString() {
        return "User{" +
                "id='" + id + '\'' +
                ", userName='" + userName + '\'' +
                ", userPwd='" + userPwd + '\'' +
                ", email='" + email + '\'' +
                ", birthday=" + birthday +
                '}';
    }
}
```

4.IUserDao

```
public interface IUserDao {

    /**
     * 根据用户名和密码来查找用户
     * @param userName
     * @param userPwd
     * @return 查到到的用户
     */
    User find(String userName, String userPwd);
```

```
    /**
     *  添加用户
     * @param user
     */
    void add(User user);

    /**根据用户名来查找用户
     * @param userName
     * @return  查到到的用户
     */
    User find(String userName);
}
```

5.UserDaoImpl

```
public class UserDaoImpl implements IUserDao {
    //要连接的数据库 URL
    private String url = "jdbc:mysql://localhost:3306/test";
    //连接的数据库时使用的用户名
    private String username = "root";
    //连接的数据库时使用的密码
    private String password = "root";
    @Override
    public User find(String userName, String userPwd) {
        try{
            //1.加载驱动
            //DriverManager.registerDriver(new com.mysql.jdbc.Driver());不推荐使用这种方式来加载驱动
            Class.forName("com.mysql.jdbc.Driver");//推荐使用这种方式来加载驱动
            //2.获取与数据库的链接
            Connection conn = DriverManager.getConnection(url, username, password);
            //3.获取用于向数据库发送 sql 语句的 statement
            Statement st = conn.createStatement();
            String sql = "select id,userName,userPwd,email,birthday from login where userName = '" + userName +"' and userPwd = '" + userPwd + "'";
            System.out.println(sql);
            //4.向数据库发 sql,并获取代表结果集的 resultset
            ResultSet rs = st.executeQuery(sql);
            User user = new User();
            //5.取出结果集的数据
            while(rs.next()){
                user.setId(String.valueOf(rs.getObject("id")));
                user.setEmail(String.valueOf(rs.getObject("email")));
                user.setUserName(String.valueOf(rs.getObject("userName")));
```

```
                user.setUserPwd(String.valueOf(rs.getObject("userPwd")));
                String birth = String.valueOf(rs.getObject("birthday"));
                SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
                user.setBirthday(sdf.parse(birth));
            }
            System.out.println(user.toString());
            return user;
        }catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    @SuppressWarnings("deprecation")
    @Override
    public void add(User user) {
        try{
            //1.加载驱动
            //DriverManager.registerDriver(new com.mysql.jdbc.Driver());不推荐使用这种方式来加载驱动
            Class.forName("com.mysql.jdbc.Driver");//推荐使用这种方式来加载驱动
            //2.获取与数据库的链接
            Connection conn = DriverManager.getConnection(url, username, password);
            String sql = "INSERT INTO login(id,userName,userPwd,email,birthday)VALUES( ?,?,?,?,?)";
            PreparedStatement pstm = conn.prepareStatement(sql);
            String birth = user.getBirthday().toString();
            SimpleDateFormat sdf1= new SimpleDateFormat("EEE MMM dd HH:mm:ss z yyyy", Locale.ENGLISH);
            SimpleDateFormat sdf2= new SimpleDateFormat("yyyy-MM-dd");
            String birthday = sdf2.format(sdf1.parse(birth));
            pstm.setString(1, user.getId());
            pstm.setString(2, user.getUserName());
            pstm.setString(3, user.getUserPwd());
            pstm.setString(4, user.getEmail());
            pstm.setString(5, birthday);
            int row = pstm.executeUpdate();
            if(row > 0){
                System.out.println("插入成功！");
            }
        }catch (Exception e) {
            System.out.println("错错错！！！");
            throw new RuntimeException(e);
        }
    }
```

```
    @Override
    public User find(String userName) {
        try{
            //1.加载驱动
            //DriverManager.registerDriver(new com.mysql.jdbc.Driver());不推荐使用这种方式来加载驱动
            Class.forName("com.mysql.jdbc.Driver");//推荐使用这种方式来加载驱动
            //2.获取与数据库的链接
            Connection conn = DriverManager.getConnection(url, username, password);
            //3.获取用于向数据库发送 sql 语句的 statement
            Statement st = conn.createStatement();
            String sql = "select id,userName,userPwd,email,birthday from login where userName = '" + userName + "'";
            //4.向数据库发 sql,并获取代表结果集的 resultset
            ResultSet rs = st.executeQuery(sql);
            User user = new User();
            //5.取出结果集的数据
            while(rs.next()){
                user.setId(String.valueOf(rs.getObject("id")));
                user.setEmail(String.valueOf(rs.getObject("email")));
                user.setUserName(String.valueOf(rs.getObject("userName")));
                user.setUserPwd(String.valueOf(rs.getObject("userPwd")));
                String birth = String.valueOf(rs.getObject("birthday"));
                SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
                user.setBirthday(sdf.parse(birth));
            }
            return user;
        }catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

}
```

6.UserExistException

```
public class UserExistException extends Exception {

    public UserExistException() {
        super();
    }

    public UserExistException(String message, Throwable cause) {
```

```
        super(message, cause);
    }

    public UserExistException(String message) {
        super(message);
    }

    public UserExistException(Throwable cause) {
        super(cause);
    }
}
```

7.IUserService

```
public interface IUserService {

    /**
     * 提供注册服务
     * @param user
     * @throws UserExistException
     */
    void registerUser(User user) throws UserExistException;

    /**
     * 提供登录服务
     * @param userName
     * @param userPwd
     * @return
     */
    User loginUser(String userName, String userPwd);
}
```

8.UserServiceImpl

```
public class UserServiceImpl implements IUserService {

    private IUserDao userDao = new UserDaoImpl();

    @Override
    public void registerUser(User user) throws UserExistException {
        User users = userDao.find(user.getUserName());
        if (users !=null && users.getUserName() != null) {
            //checked exception
```

```
                //unchecked exception
                //这里抛编译时异常的原因：是我想上一层程序处理这个异常，以给用户一个友好提示
                throw new UserExistException("注册的用户名已存在！！！");
            }
            userDao.add(user);
        }

        @Override
        public User loginUser(String userName, String userPwd) {
            return userDao.find(userName, userPwd);
        }

}
```

9.WebUtils

```
public class WebUtils {

    /**
     * 将 request 对象转换成 T 对象
     * @param request
     * @param clazz
     * @return
     */
    public static <T> T request2Bean(HttpServletRequest request,Class<T> clazz){
        try{
            T bean = clazz.newInstance();
            Enumeration<String> e = request.getParameterNames();
            while(e.hasMoreElements()){
                String name = (String) e.nextElement();
                String value = request.getParameter(name);
                BeanUtils.setProperty(bean, name, value);
            }
            return bean;
        }catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    /**
     * 生成 UUID
     * @return
     */
```

```
    public static String makeId(){
        return UUID.randomUUID().toString();
    }

}
```

10.CharacterEncodingFilter

```
public class CharacterEncodingFilter implements Filter {

    //存储系统使用的字符编码
    private String encoding=null;

    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
        //encoding 在 web.xml 中指定
        this.encoding = filterConfig.getInitParameter("encoding");
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
            FilterChain chain) throws IOException, ServletException {
        //解决表单提交时的中文乱码问题
        request.setCharacterEncoding(encoding);
        chain.doFilter(request, response);
    }

    @Override
    public void destroy() {

    }
}
```

11.RegisterFormBean

```
/**
 * 封装的用户注册表单 bean，用来接收 register.jsp 中的表单输入项的值
 * RegisterFormBean 中的属性与 register.jsp 中的表单输入项的 name 一一对应
 * RegisterFormBean 的职责除了负责接收 register.jsp 中的表单输入项的值之外还担任着校验表单输入项的值的合法性
 * @author gacl
 *
 */
```

```
public class RegisterFormBean {

	//RegisterFormBean 中的属性与 register.jsp 中的表单输入项的 name 一一对应
	//<input type="text" name="userName"/>
	private String userName;
	//<input type="password" name="userPwd"/>
	private String userPwd;
	//<input type="password" name="confirmPwd"/>
	private String confirmPwd;
	//<input type="text" name="email"/>
	private String email;
	//<input type="text" name="birthday"/>
	private String birthday;



	/**
	 * 存储校验不通过时给用户的错误提示信息
	 */
	private Map<String, String> errors = new HashMap<String, String>();

	public Map<String, String> getErrors() {
		return errors;
	}

	public void setErrors(Map<String, String> errors) {
		this.errors = errors;
	}

	/*
	 * validate 方法负责校验表单输入项
	 * 表单输入项校验规则：
	 *		private String userName; 用户名不能为空，并且要是 3-8 的字母 abcdABcd
	 *		private String userPwd; 密码不能为空，并且要是 3-8 的数字
	 *		private String confirmPwd; 两次密码要一致
	 *		private String email; 可以为空，不为空要是一个合法的邮箱
	 *		private String birthday; 可以为空，不为空时，要是一个合法的日期
	 */
	public boolean validate() {

		boolean isOk = true;

		if (this.userName == null || this.userName.trim().equals("")) {
			isOk = false;
			errors.put("userName", "用户名不能为空！！");
```

```
} else {
        if (!this.userName.matches("[a-zA-Z]{3,8}")) {
                isOk = false;
                errors.put("userName", "用户名必须是 3-8 位的字母！！");
        }
}

if (this.userPwd == null || this.userPwd.trim().equals("")) {
        isOk = false;
        errors.put("userPwd", "密码不能为空！！");
} else {
        if (!this.userPwd.matches("\\d{3,8}")) {
                isOk = false;
                errors.put("userPwd", "密码必须是 3-8 位的数字！！");
        }
}

// private String password2; 两次密码要一致
if (this.confirmPwd != null) {
        if (!this.confirmPwd.equals(this.userPwd)) {
                isOk = false;
                errors.put("confirmPwd", "两次密码不一致！！");
        }
}

// private String email; 可以为空，不为空要是一个合法的邮箱
if (this.email != null && !this.email.trim().equals("")) {
        if (!this.email.matches("\\w+@\\w+(\\.\\w+)+")) {
                isOk = false;
                errors.put("email", "邮箱不是一个合法邮箱！！");
        }
}

// private String birthday; 可以为空，不为空时，要是一个合法的日期
if (this.birthday != null && !this.birthday.trim().equals("")) {
        try {
                DateLocaleConverter conver = new DateLocaleConverter();
                conver.convert(this.birthday);
        } catch (Exception e) {
                isOk = false;
                errors.put("birthday", "生日必须要是一个日期！！");
        }
}
```

```
        return isOk;
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getUserPwd() {
        return userPwd;
    }

    public void setUserPwd(String userPwd) {
        this.userPwd = userPwd;
    }

    public String getConfirmPwd() {
        return confirmPwd;
    }

    public void setConfirmPwd(String confirmPwd) {
        this.confirmPwd = confirmPwd;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getBirthday() {
        return birthday;
    }

    public void setBirthday(String birthday) {
        this.birthday = birthday;
    }
}
```

12.index.jsp

```
<%@ page language="java"    pageEncoding="UTF-8"%>
<%--为了避免在 jsp 页面中出现 java 代码，这里引入 jstl 标签库，利用 jstl 标签库提供的标签来做一些逻辑判断处理  --%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE HTML>
<html>
<head>
    <title>首页</title>
    <script type="text/javascript">
        function doLogout(){
            //访问 LogoutServlet 注销当前登录的用户

window.location.href="${pageContext.request.contextPath}/servlet/LogoutServlet";
        }
    </script>
</head>

<body>
<h1>何其有静的网站</h1>
<hr/>
<c:if test="${user==null}">
    <a href="${pageContext.request.contextPath}/servlet/RegisterUIServlet" target="_blank">注册</a>
    <a href="${pageContext.request.contextPath}/servlet/LoginUIServlet">登陆</a>
</c:if>
<c:if test="${user!=null}">
    欢迎您：${user.userName}
    <input type="button" value="退出登陆" onclick="doLogout()">
</c:if>
<hr/>
</body>
</html>
```

13.web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://java.sun.com/xml/ns/javaee"
         xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
         http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd" version="3.0">
```

```
<display-name></display-name>
<servlet>
    <servlet-name>LoginUIServlet</servlet-name>
    <servlet-class>me.gacl.web.UI.LoginUIServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>RegisterUIServlet</servlet-name>
    <servlet-class>me.gacl.web.UI.RegisterUIServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>RegisterServlet</servlet-name>
    <servlet-class>me.gacl.web.controller.RegisterServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>LoginServlet</servlet-name>
    <servlet-class>me.gacl.web.controller.LoginServlet</servlet-class>
</servlet>
<servlet>
    <servlet-name>LogoutServlet</servlet-name>
    <servlet-class>me.gacl.web.controller.LogoutServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>LoginUIServlet</servlet-name>
    <url-pattern>/servlet/LoginUIServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>RegisterUIServlet</servlet-name>
    <url-pattern>/servlet/RegisterUIServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>RegisterServlet</servlet-name>
    <url-pattern>/servlet/RegisterServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>LoginServlet</servlet-name>
    <url-pattern>/servlet/LoginServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>LogoutServlet</servlet-name>
    <url-pattern>/servlet/LogoutServlet</url-pattern>
</servlet-mapping>
<filter>
    <filter-name>EncodingFilter</filter-name>
    <filter-class>me.gacl.web.filter.CharacterEncodingFilter</filter-class>
```

```
        <init-param>
            <param-name>encoding</param-name>
            <param-value>UTF-8</param-value>
        </init-param>
    </filter>
    <filter-mapping>
        <filter-name>EncodingFilter</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

14.RegisterUIServlet

```
/**
 * @author gacl
 * 为用户提供注册的用户界面的 Servlet
 * RegisterUIServlet 负责为用户输出注册界面
 * 当用户访问 RegisterUIServlet 时，就跳转到 WEB-INF/pages 目录下的 register.jsp 页面
 */
public class RegisterUIServlet extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        request.getRequestDispatcher("/WEB-INF/pages/register.jsp").forward(request,
response);
    }

    public void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        doGet(request, response);
    }

}
```

15.register.jsp

```
<%@ page language="java" pageEncoding="UTF-8"%>
<!DOCTYPE HTML>
<html>
<head>
```

```
    <title>用户注册</title>
</head>

<body style="text-align: center;">
<form action="${pageContext.request.contextPath}/servlet/RegisterServlet" method="post">
    <table width="60%" border="1">
        <tr>
            <td>用户名</td>
            <td>
                <%--使用 EL 表达式${}提取存储在 request 对象中的 formbean 对象中封装的表单数据(formbean.userName)以及错误提示消息(formbean.errors.userName)--%>
                <input type="text" name="userName" value="${formbean.userName}">${formbean.errors.userName}
            </td>
        </tr>
        <tr>
            <td>密码</td>
            <td>
                <input type="password" name="userPwd" value="${formbean.userPwd}">${formbean.errors.userPwd}
            </td>
        </tr>
        <tr>
            <td>确认密码</td>
            <td>
                <input type="password" name="confirmPwd" value="${formbean.confirmPwd}">${formbean.errors.confirmPwd}
            </td>
        </tr>
        <tr>
            <td>邮箱</td>
            <td>
                <input type="text" name="email" value="${formbean.email}">${formbean.errors.email}
            </td>
        </tr>
        <tr>
            <td>生日</td>
            <td>
                <input type="text" name="birthday" value="${formbean.birthday}">${formbean.errors.birthday}
            </td>
        </tr>
        <tr>
```

```
                <td>
                    <input type="reset" value="清空">
                </td>
                <td>
                    <input type="submit" value="注册">
                </td>
            </tr>
        </table>
</form>
</body>
</html>
```

16.RegisterServlet

```
/**
 * 处理用户注册的 Servlet
 * @author gacl
 *
 */
public class RegisterServlet extends HttpServlet {

    private Logger logger = Logger.getLogger(RegisterServlet.class.getName());

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        //将客户端提交的表单数据封装到 RegisterFormBean 对象中
        RegisterFormBean formbean = WebUtils.request2Bean(request,RegisterFormBean.class);
        //校验用户注册填写的表单数据
        if (formbean.validate() == false) {//如果校验失败
            //将封装了用户填写的表单数据的 formbean 对象发送回 register.jsp 页面的 form 表单中进行显示
            request.setAttribute("formbean", formbean);
            //校验失败就说明是用户填写的表单数据有问题，那么就跳转回 register.jsp
            request.getRequestDispatcher("/WEB-INF/pages/register.jsp").forward(request, response);
            return;
        }

        User user = new User();
        try {
            // 注册字符串到日期的转换器
            ConvertUtils.register(new DateLocaleConverter(), Date.class);
            BeanUtils.copyProperties(user, formbean);//把表单的数据填充到 javabean 中
```

```
                user.setId(WebUtils.makeId());//设置用户的 Id 属性
                IUserService service = new UserServiceImpl();
                //调用 service 层提供的注册用户服务实现用户注册
                service.registerUser(user);
                String message = String.format(
                        "注册成功！！3 秒后为您自动跳到登录页面！！ <meta
http-equiv='refresh' content='3;url=%s'/>",
                        request.getContextPath()+"/servlet/LoginUIServlet");
                request.setAttribute("message",message);
                request.getRequestDispatcher("/message.jsp").forward(request,response);

            } catch (UserExistException e) {
                formbean.getErrors().put("userName", "注册用户已存在！！");
                request.setAttribute("formbean", formbean);
                request.getRequestDispatcher("/WEB-INF/pages/register.jsp").forward(request,
response);
            } catch (Exception e) {
                e.printStackTrace(); // 在后台记录异常
                request.setAttribute("message", "对不起，注册失败！！");
                request.getRequestDispatcher("/message.jsp").forward(request,response);
            }
        }

        public void doPost(HttpServletRequest request, HttpServletResponse response)
                throws ServletException, IOException {
            doGet(request, response);
        }

}
```

17.message.jsp

```
<%@ page language="java" pageEncoding="UTF-8"%>
<!DOCTYPE HTML>
<html>
  <head>
    <title>全局消息显示页面</title>
  </head>

  <body>
    ${message}
  </body>
</html>
```

18.LoginUIServlet

```
/**
 * @author gacl
 * LoginUIServlet 负责为用户输出登陆界面
 *  当用户访问 LoginUIServlet 时，就跳转到 WEB-INF/pages 目录下的 login.jsp 页面
 */
public class LoginUIServlet extends HttpServlet {

	public void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {

		request.getRequestDispatcher("/WEB-INF/pages/login.jsp").forward(request,
response);
	}

	public void doPost(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		doGet(request, response);
	}

}
```

19.login.jsp

```
/**
 * @author gacl
 * LoginUIServlet 负责为用户输出登陆界面
 *  当用户访问 LoginUIServlet 时，就跳转到 WEB-INF/pages 目录下的 login.jsp 页面
 */
public class LoginUIServlet extends HttpServlet {

	public void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {

		request.getRequestDispatcher("/WEB-INF/pages/login.jsp").forward(request,
response);
	}

	public void doPost(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		doGet(request, response);
	}
```

```
}
```

20.LoginServlet

```
/**
 * 处理用户登录的 servlet
 * @author gacl
 *
 */
public class LoginServlet extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        //获取用户填写的登录用户名
        String username = request.getParameter("username");
        //获取用户填写的登录密码
        String password = request.getParameter("password");
        IUserService service = new UserServiceImpl();
        //用户登录
        User user = service.loginUser(username, password);
        if(user==null || user.getUserName() == null){
            String message = String.format(
                    "对不起，用户名或密码有误！！请重新登录！2 秒后为您自动跳到登录页面！！<meta http-equiv='refresh' content='2;url=%s'",
                    request.getContextPath()+"/servlet/LoginUIServlet");
            request.setAttribute("message",message);
            request.getRequestDispatcher("/message.jsp").forward(request, response);
            return;
        }
        //登录成功后，就将用户存储到 session 中
        request.getSession().setAttribute("user", user);
        String message = String.format(
                "恭喜：%s,登陆成功!本页将在 3 秒后跳到首页!!<meta http-equiv='refresh' content='3;url=%s'",
                user.getUserName(),
                request.getContextPath()+"/index.jsp");
        request.setAttribute("message",message);
        request.getRequestDispatcher("/message.jsp").forward(request, response);
    }

    public void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
```

```
        doGet(request, response);
    }

}
```

21.LogoutServlet

```
public class LogoutServlet extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        //移除存储在 session 中的 user 对象，实现注销功能
        request.getSession().removeAttribute("user");
        //由于字符串中包含有单引号，在这种情况下使用 MessageFormat.format 方法拼接字符串时就会有问题
        //MessageFormat.format 方法只是把字符串中的单引号去掉，不会将内容填充到指定的占位符中
        String tempStr1 = MessageFormat.format(
                "注销成功！！3 秒后为您自动跳到登录页面！！<meta http-equiv='refresh' content='3;url={0}'/>",
                request.getContextPath()+"/servlet/LoginUIServlet");
        System.out.println(tempStr1);//输出结果：注销成功！！3 秒后为您自动跳到登录页面！！<meta http-equiv=refresh content=3;url={0}/>
        System.out.println("------------------------------------------------------------");
        /**
         * 要想解决"如果要拼接的字符串包含有单引号，那么 MessageFormat.format 方法就只是把字符串中的单引号去掉，不会将内容填充到指定的占位符中"这个问题，
         * 那么可以需要使用单引号引起来的字符串中使用 2 个单引号引起来，例如："<meta http-equiv=''refresh'' content=''3;url={0}''/>"
         * 这样 MessageFormat.format("<meta http-equiv=''refresh'' content=''3;url={0}''/>","index.jsp")就可以正常返回
         * <meta http-equiv=''refresh'' content=''3;url=index.jsp'/>
         */
        String tempStr2 = MessageFormat.format(
                "注销成功！！3 秒后为您自动跳到登录页面！！<meta http-equiv=''refresh'' content=''3;url={0}''/>",
                request.getContextPath()+"/servlet/LoginUIServlet");
        /**
         * 输出结果：
         * 注销成功！！3 秒后为您自动跳到登录页面！！
         * <meta http-equiv='refresh' content='3;url=/webmvcframework/servlet/LoginUIServlet'/>
         */
        System.out.println(tempStr2);
```

```
            String message = String.format(
                        "注销成功！！3 秒后为您自动跳到登录页面！！<meta http-equiv='refresh'
content='3;url=%s'/>",
                        request.getContextPath()+"/servlet/LoginUIServlet");
            request.setAttribute("message",message);
            request.getRequestDispatcher("/message.jsp").forward(request, response);
      }

      public void doPost(HttpServletRequest request, HttpServletResponse response)
                  throws ServletException, IOException {
            doGet(request, response);
      }

}
```

22.项目结构图(maven)