



STRUTS1

STRUTS2

SPRING

IBATIS

MYBATIS

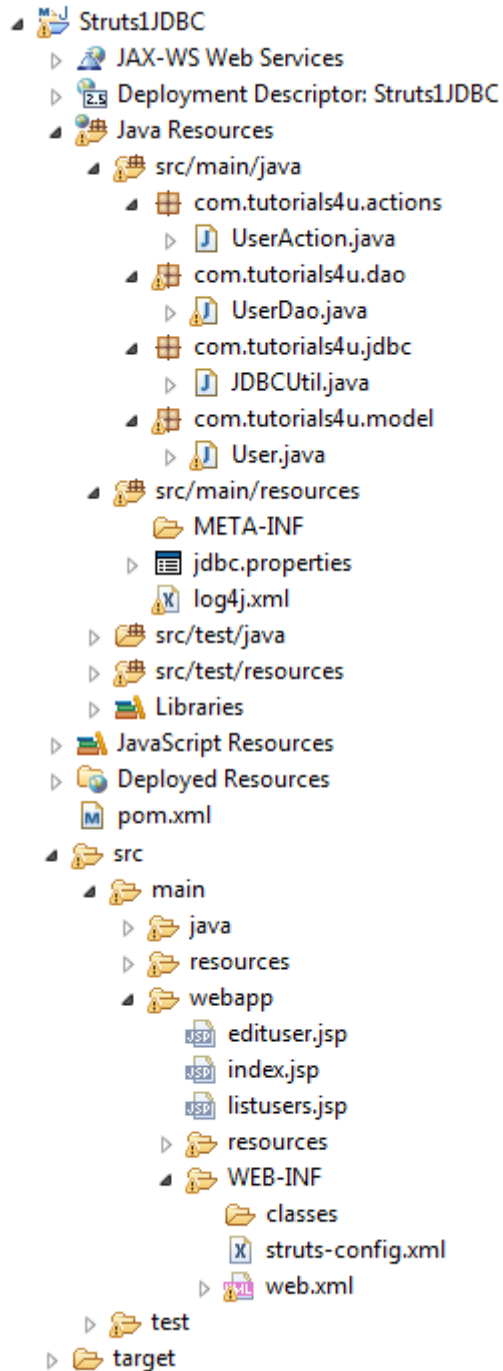
HIBERNATE

INTEGRATIONS

Sr No	Description	Page No
1	Struts1 JDBC Integration	2
2	Struts1 Hibernate Integration	16
3	Struts1 iBatis Integration	26
4	Struts1 Mybatis Integration	36
5	Struts1 Spring JDBC Integration	47
6	Struts1 Spring Hibernate Integration	59
7	Struts1 Spring iBatis Integration	71
8	Struts1 Spring MyBatis Integration	83
9	Struts2 JDBC Integration	95
10	Struts2 Hibernate Integration	118
11	Struts2 iBatis Integration	128
12	Struts2 MyBatis Integration	138
13	Struts2 Spring JDBC Integration	150
14	Struts2 Spring Hibernate Integration	165
15	Struts2 Spring iBatis Integration	177
16	Struts2 Spring Mybatis Integration	189
17	Spring JDBCTemplate Integration	202
18	Spring Hibernate Integration	
	1) Using XML Mapping Configuration	217
	2) Using Annotations	232
19	Spring iBatis Integration	248
20	Spring Mybatis Integration	
	1) Using XML Mapping Configuration	262
	2) Using Annotations	278

➤ Struts1 JDBC Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
    <display-name>Struts1 JDBC Integration</display-name>

    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
            <param-value>/WEB-INF/struts-config.xml</param-value>
        </init-param>
        <load-on-startup>2</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>action</servlet-name>
        <url-pattern>*.do</url-pattern>
```

```
</servlet-mapping>

<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
  "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
  "http://struts.apache.org/dtds/struts-config_1_3.dtd">
<struts-config>
  <form-beans>
    <form-bean name="user" type="com.tutorials4u.model.User" />
  </form-beans>
  <action-mappings>
    <action path="/helloWorld"
      type="org.apache.struts.actions.ForwardAction"
      parameter="/index.jsp">
    </action>

    <action path="/insertUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/index.jsp" parameter="insertUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/listUsers" type="com.tutorials4u.actions.UserAction"
      parameter="listUsers">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/deleteUser" type="com.tutorials4u.actions.UserAction"
      parameter="deleteUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser" type="com.tutorials4u.actions.UserAction"
      parameter="editUser">
      <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/edituser.jsp" parameter="updateUser">
      <forward name="success" path="/listusers.jsp" />
    </action>
  </action-mappings>
</struts-config>
```

Jdbc.properties:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file"
            value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log" />
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
import javax.persistence.Transient;
import org.apache.struts.action.ActionForm;

public class User extends ActionForm{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
```

```
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }
}
```

```
@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserDao userDao = new UserDao();
    public ActionForward insertUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.insertUser()");
        String result = userDao.insertUser(user);
        logger.debug("1 Record Inserted Successfully " + result);
        return mapping.findForward("success");
    }

    public ActionForward listUsers(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.listUsers()");
        List<User> users = userDao.listUsers();
        request.setAttribute("users", users);
        return mapping.findForward("success");
    }

    public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.deleteUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        logger.debug("value="+list.get(1));
        userDao.deleteUser(Integer.parseInt(list.get(1)));
        logger.debug("1 Record Deleted Successfully ");
    }
}
```



```
        return mapping.findForward("success");
    }

    public ActionForward editUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        Logger.debug("UserAction.editUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        User user=userDao.editUser(Integer.parseInt(list.get(1)));
        request.setAttribute("user",user);
        return mapping.findForward("success");
    }

    public ActionForward updateUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        Logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
        Logger.debug("1 Record Updated Successfully");
        return mapping.findForward("success");
    }
}
```

JDBCUtil.java:

```
package com.tutorials4u.jdbc;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.util.Properties;

public class JDBCUtil {
    private static Connection connection = null;
    InputStream inputStream;
    String propFileName = "jdbc.properties";
    public static Properties prop = new Properties();
    public JDBCUtil() throws IOException {
        inputStream = getClass().getClassLoader().getResourceAsStream(
            propFileName);
        if (inputStream != null) {
            prop.load(inputStream);
        } else {
            throw new FileNotFoundException("property file '" +
propFileName
            + "' not found in the classpath");
        }
    }

    public static Connection getMysqlConnection() throws IOException {
        String driverClass = prop.getProperty("jdbc.driverClassName");
```

```
String url = prop.getProperty("jdbc.url");
String username = prop.getProperty("jdbc.username");
String password = prop.getProperty("jdbc.password");

if (connection == null) {
    try {
        Class.forName(driverClass).newInstance();
        connection = DriverManager.getConnection(url, username,
            password);
        System.out.println("Connected To Mysql Datasbase");
    } catch (Exception e) {
        e.printStackTrace();
    }
}
return connection;
}
}
```

UserDao.java:

```
package com.tutorials4u.dao;
import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.jdbc.JDBCUtil;
import com.tutorials4u.model.User;

public class UserDao {
    private static final Logger logger = Logger.getLogger(UserAction.class);

    PreparedStatement stmt = null;

    @SuppressWarnings("static-access")
    public String insertUser(User newuser) throws IOException, SQLException {
        logger.debug("UserDao.createUser()");
        JDBCUtil util = new JDBCUtil();
        Connection conn = util.getMysqlConnection();
        try {
            String username = newuser.getUsername();
            String password = newuser.getPassword();
            String firstname = newuser.getFirstName();
            String lastname = newuser.getLastName();
            String emailId = newuser.getEmailId();
            String phone = newuser.getPhone();

            String sql = new String(
                "insert into myuser(USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID) values(?,?,?,?,?,?)");

            stmt = conn.prepareStatement(sql);
```

```
        stmt.setString(1, username);
        stmt.setString(2, password);
        stmt.setString(3, firstname);
        stmt.setString(4, lastname);
        stmt.setString(5, phone);
        stmt.setString(6, emailId);
        logger.debug("Inserting values....");
        stmt.executeUpdate();
    } catch (Exception e) {
        logger.error("Error creating User :" + e.getMessage());
        return "failure";
    } finally {
        stmt.close();
        conn.close();
    }
    return "success";
}

@SuppressWarnings("static-access")
public List<User> listUsers() throws IOException, SQLException {
    logger.debug("UserDao.listUsers()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    List<User> users = new ArrayList<User>();
    try {
        String sql = "select
USER_ID,USERNAME,FIRSTNAME,LASTNAME,PHONE,EMAILID from
myuser";
        statement = conn.createStatement();
        result = statement.executeQuery(sql);
        User newuser = null;
        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
            users.add(newuser);
        }
    } catch (SQLException e) {
        logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
        // conn.close();
    }
    return users;
}
```

```
@SuppressWarnings("static-access")
public void deleteUser(int id) throws IOException, SQLException {
    Logger.debug("UserDao.deleteUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    Statement statement = null;
    try {
        String sql = "delete from myuser where user_id=" + id;
        statement = conn.createStatement();
        statement.executeUpdate(sql);
    } catch (SQLException e) {
        Logger.error("Error deleting User :" + e.getMessage());
    } finally {
        statement.close();
    }
}

@SuppressWarnings("static-access")
public User editUser(int id) throws IOException, SQLException {
    Logger.debug("UserDao.editUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    User user = getUser(id);
    return user;
}

@SuppressWarnings("static-access")
public User getUser(int id) throws SQLException, IOException {
    Logger.debug("UserDao.getUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    User newuser = null;
    try {
        String sql = "select
USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
from myuser WHERE USER_ID="+ id;
        statement = conn.createStatement();
        result = statement.executeQuery(sql);

        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setPassword(result.getString("PASSWORD"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
        }
    } catch (SQLException e) {
        Logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
    }
}
```

```
        return newuser;
    }
    @SuppressWarnings("static-access")
    public void updateUser(User newuser) throws IOException {
        Logger.debug("UserDao.updateUser()");
        JDBCUtil util = new JDBCUtil();
        Connection conn = util.getMysqlConnection();
        try {
            int userid = newuser.getUser_id();
            String username = newuser.getUsername();
            String password = newuser.getPassword();
            String firstname = newuser.getFirstName();
            String lastname = newuser.getLastName();
            String emailId = newuser.getEmailId();
            String phone = newuser.getPhone();

            String sql = "UPDATE MYUSER SET
            USERNAME=?,PASSWORD=?,FIRSTNAME=?,LASTNAME=?,PHONE=?,EMAILID=?
            WHERE USER_ID="+userid;

            stmt = conn.prepareStatement(sql);
            stmt.setString(1, username);
            stmt.setString(2, password);
            stmt.setString(3, firstname);
            stmt.setString(4, lastname);
            stmt.setString(5, phone);
            stmt.setString(6, emailId);

            Logger.debug("Updating values....");
            stmt.executeUpdate();
        } catch (Exception e) {
            Logger.error("Error updating User :" + e.getMessage());
        }
    }
}
```

Index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@taglib uri="http://struts.apache.org/tags-bean" prefix="bean"%>
<%@taglib uri="http://struts.apache.org/tags-logic" prefix="logic"%>
<%@taglib uri="http://struts.apache.org/tags-html" prefix="html"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert User Details</title>
</head>
<body id="dt_example">
    <b>Struts1 + JDBC Integration</b>
    <div class="full_w">

        <html:form action="/insertUser">
            <div class="element">
                <html:hidden property="user_id" />
```

```
</div>

<div class="element">
  <label for="name"><span class="red">*</span>First
  Name</label>
  <html:text property="firstName"
    onkeydown="return isAlpha(event.keyCode);" />
</div>

<div class="element">
  <label for="name"><span class="red">*</span>Last
  Name</label>
  <html:text property="lastName"
    onkeydown="return isAlpha(event.keyCode);" />
</div>

<div class="element">
  <label for="name"><span
    class="red">*</span>Username</label>
  <html:text property="username" />
</div>

<div class="element">
  <label for="name"><span
    class="red">*</span>Password</label>
  <html:password property="password" />
</div>

<div class="element">
  <label for="name"><span
    class="red">*</span>Phone</label>
  <html:text property="phone" onkeypress="return
    isNumberKey(event)" maxlength="10" />
</div>

<div class="element">
  <label for="name"><span class="red">*</span>Email Id</label>
  <html:text property="emailId" />
</div>

<div class="entry">
  <html:submit>Save</html:submit>
</div>
</html:form>
</div>
</body>
</html>
```

listUsers.jsp:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@taglib uri="http://struts.apache.org/tags-bean" prefix="bean"%>
<%@taglib uri="http://struts.apache.org/tags-logic" prefix="logic"%>
<%@taglib uri="http://struts.apache.org/tags-html" prefix="html"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
```

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>List Of Users</title>
<table id="companies" class="display">
    <thead>
        <tr>
            <th>Username</th>
            <th><u>First Name</u></th>
            <th><u>Last Name</u></th>
            <th><u>Phone</u></th>
            <th><u>Email Id</u></th>
            <th>Actions</th>
        </tr>
    </thead>
    <tbody>
        <c:forEach items="${users}" var="user">
            <tr>
                <td align='center'><c:out value="${user.username}" /></td>
                <td align='center'><c:out value="${user.firstName}" /></td>
                <td align='center'><c:out value="${user.lastName}" /></td>
                <td align='center'><c:out value="${user.phone}" /></td>
                <td align='center'><c:out value="${user.emailId}" /></td>
                <td align='center'><a
                    href="/Struts1Hibernate/editUser.do?userid=${user.user_id}"></a>&nbsp;&nbsp;&nbsp;<a
                        href="/Struts1Hibernate/deleteUser.do?userid=${user.user_id}"></a></td>
            </tr>
        </c:forEach>
    </tbody>
</table>
</div>
</body>
</html>
```

edituser.jsp:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@taglib uri="http://struts.apache.org/tags-bean" prefix="bean"%>
<%@taglib uri="http://struts.apache.org/tags-logic" prefix="logic"%>
<%@taglib uri="http://struts.apache.org/tags-html" prefix="html"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Edit User Details</title>
</head>
<body id="dt_example">
    <b>Struts1 + JDBC Integration</b>
    <br>
    <br>

    <div class="full_w">
```

```
<html:form action="/updateUser">
  <div class="element">
    <html:hidden property="user_id" value="${user.user_id}"/>
  </div>

  <div class="element">
    <label for="name"><span class="red">*</span>First Name</label>
    <html:text property="firstName"
      onkeydown="return isAlpha(event.keyCode);"
      value="${user.firstName}"/>
  </div>

  <div class="element">
    <label for="name"><span class="red">*</span>Last Name</label>
    <html:text property="lastName"
      onkeydown="return isAlpha(event.keyCode);"
      value="${user.lastName}"/>
  </div>

  <div class="element">
    <label for="name"><span class="red">*</span>Username</label>
    <html:text property="username" value="${user.username}"
      readonly="true"/>
  </div>

  <div class="element">
    <label for="name"><span class="red">*</span>Password</label>
    <html:password property="password"
      value="${user.password}"/>
  </div>

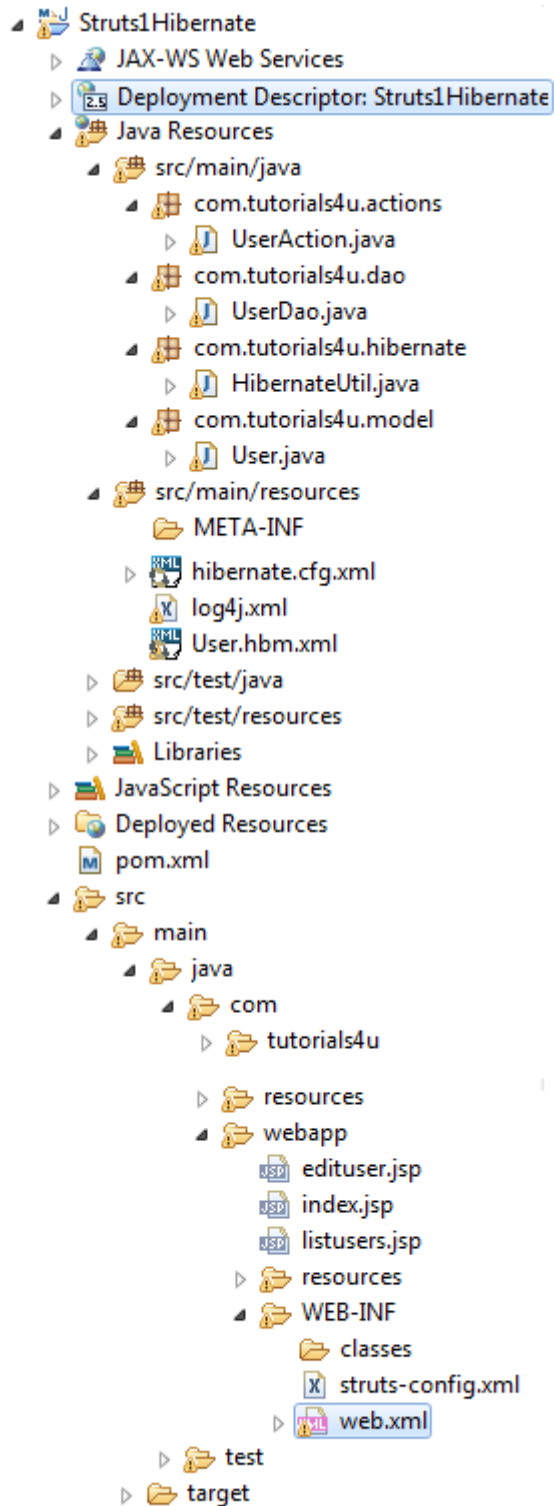
  <div class="element">
    <label for="name"><span class="red">*</span>Phone</label>
    <html:text property="phone" onkeypress="return
      isNumberKey(event)" maxlength="10"
      value="${user.phone}"/>
  </div>

  <div class="element">
    <label for="name"><span class="red">*</span>Email
      Id</label>
    <html:text property="emailId" value="${user.emailId}"/>
  </div>

  <div class="entry">
    <html:submit>Save</html:submit>
  </div>
</html:form>
</div>
</body>
</html>
```


➤ Struts1 Hibernate Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Hibernate framework -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>4.2.4.Final</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
    <display-name>Struts1 Hibernate Integration</display-name>

    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
```

```
<param-value>/WEB-INF/struts-config.xml</param-value>
</init-param>
<load-on-startup>2</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>

<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
  "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
  "http://struts.apache.org/dtds/struts-config_1_3.dtd">
<struts-config>
  <form-beans>
    <form-bean name="user" type="com.tutorials4u.model.User" />
  </form-beans>
  <action-mappings>
    <action path="/helloWorld"
type="org.apache.struts.actions.ForwardAction" parameter="/index.jsp">
    </action>

    <action path="/insertUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/index.jsp" parameter="insertUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/listUsers" type="com.tutorials4u.actions.UserAction"
      parameter="listUsers">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/deleteUser" type="com.tutorials4u.actions.UserAction"
      parameter="deleteUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser" type="com.tutorials4u.actions.UserAction"
      parameter="editUser">
      <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/edituser.jsp" parameter="updateUser">
      <forward name="success" path="/listusers.jsp" />
    </action>
  </action-mappings>
</struts-config>
```

Hibernate.cfg.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property
name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/hibernate</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password"></property>
        <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="show_sql">>true</property>
        <property name="format_sql">>true</property>
        <property name="hbm2ddl.auto">update</property>
        <mapping resource="User.hbm.xml" />
    </session-factory>
</hibernate-configuration>
```

User.hbm.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.tutorials4u.model.User" table="MYUSER">
        <meta attribute="class-description">
            This class contains the User detail.
        </meta>
        <id name="user_id" type="int" column="USER_ID">
            <generator class="native" />
        </id>
        <property name="username" column="USERNAME" type="string" />
        <property name="password" column="PASSWORD" type="string" />
        <property name="firstName" column="FIRSTNAME" type="string" />
        <property name="lastName" column="LASTNAME" type="string" />
        <property name="phone" column="PHONE" type="string" />
        <property name="emailId" column="EMAILID" type="string" />
    </class>
</hibernate-mapping>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
```

```
        value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
</appender>

<!-- file -->
<appender name="file" class="org.apache.log4j.RollingFileAppender">
    <param name="append" value="false" />
    <param name="maxFileSize" value="10KB" />
    <param name="maxBackupIndex" value="5" />
    <param name="file"
value="F:/Coding/StrutsCoding/Struts1.x/logs/Logs.log" />
    <layout class="org.apache.log4j.PatternLayout">
        <param name="ConversionPattern"
value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
</appender>

<root>
    <level value="DEBUG" />
    <appender-ref ref="console" />
    <appender-ref ref="file" />
</root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
import javax.persistence.Transient;
import org.apache.struts.action.ActionForm;
```

```
public class User extends ActionForm{

    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }
}
```

```
public String getUsername() {
    return username;
}

public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserDao userDao = new UserDao();

    public ActionForward insertUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.insertUser()");
        String result = userDao.insertUser(user);
        logger.debug("1 Record Inserted Successfully " + result);
        return mapping.findForward("success");
    }

    public ActionForward listUsers(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.listUsers()");
        List<User> users = userDao.listUsers();
        request.setAttribute("users", users);
        return mapping.findForward("success");
    }

    public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.deleteUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        logger.debug("value="+list.get(1));
        userDao.deleteUser(Integer.parseInt(list.get(1)));
        logger.debug("1 Record Deleted Successfully ");
        return mapping.findForward("success");
    }
}
```

```
public ActionForward editUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    logger.debug("UserAction.editUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    User user=userDao.editUser(Integer.parseInt(list.get(1)));
    request.setAttribute("user",user);
    return mapping.findForward("success");
}

public ActionForward updateUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    User user = (User) form;
    logger.debug("UserAction.updateUser()");
    userDao.updateUser(user);
    logger.debug("1 Record Updated Successfully");
    return mapping.findForward("success");
}
}
```

HibernateUtil.java:

```
package com.tutorials4u.hibernate;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class HibernateUtil {

    private static final SessionFactory sessionFactory =
        buildSessionFactory();

    private static SessionFactory buildSessionFactory() {
        try {
            // Create the SessionFactory from hibernate.cfg.xml
            return new Configuration().configure().buildSessionFactory();
        }
        catch (Throwable ex) {
            // Make sure you log the exception, as it might be swallowed
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

    public static void shutdown() {
        // Close caches and connection pools
        getSessionFactory().close();
    }
}
```


UserDao.java:

```
package com.tutorials4u.dao;
import java.util.List;
import org.hibernate.Session;
import org.apache.log4j.Logger;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.hibernate.HibernateUtil;
import com.tutorials4u.model.User;

public class UserDao {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);

    public String insertUser(User newUser) {
        logger.debug("UserDao.createUser()");
        Session session = HibernateUtil.getSessionFactory().openSession();
        try {
            session.beginTransaction();
            User user = new User();
            user.setUsername(newUser.getUsername());
            user.setPassword(newUser.getPassword());
            user.setFirstName(newUser.getFirstName());
            user.setLastName(newUser.getLastName());
            user.setEmailId(newUser.getEmailId());
            user.setPhone(newUser.getPhone());
            logger.debug("Inserting values....");
            session.save(user);
            session.getTransaction().commit();
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            session.getTransaction().rollback();
            return "failure";
        }
        return "success";
    }

    public List<User> listUsers() {
        logger.debug("UserDao.listUsers()");
        Session session = HibernateUtil.getSessionFactory().openSession();
        return session.createQuery("From User").list();
    }

    public void deleteUser(int id) {
        logger.debug("UserDao.deleteUser()");
        Session session = HibernateUtil.getSessionFactory().openSession();
        session.beginTransaction();
        User user = getUser(id);
        session.delete(user);
        session.getTransaction().commit();
    }

    public User editUser(int id) {
        logger.debug("UserDao.editUser()");
        Session session = HibernateUtil.getSessionFactory().openSession();
        User user = getUser(id);
        return user;
    }
}
```

```
public User getUser(int id) {
    logger.debug("UserDao.getUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    User user = (User) session.get(User.class, id);
    return user;
}

public void updateUser(User user) {
    logger.debug("UserDao.updateUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.update(user);
    session.getTransaction().commit();
}
}
```

Index.jsp, edituser.jsp, listUsers.jsp are same as above Struts1 JDBC Integration

➤ Struts1 iBatis Integration

Project Structure:

- ▲ Struts1iBatis
 - ▶ JAX-WS Web Services
 - ▶ Deployment Descriptor: Struts1iBatis
 - ▲ Java Resources
 - ▲ src/main/java
 - ▲ com.tutorials4u.actions
 - ▶ UserAction.java
 - ▲ com.tutorials4u.dao
 - ▶ UserDao.java
 - ▲ com.tutorials4u.ibatis
 - ▶ IbatisUtil.java
 - ▶ com.tutorials4u.model
 - ▲ src/main/resources
 - META-INF
 - log4j.xml
 - sql-maps-config.xml
 - user.xml
 - ▲ src/test/java
 - ▶ com.tutorials4u.test
 - ▶ src/test/resources
 - ▶ Libraries
 - ▶ JavaScript Resources
 - ▶ Deployed Resources
 - pom.xml
 - ▲ src
 - ▲ main
 - ▶ java
 - ▶ resources
 - ▲ webapp
 - edituser.jsp
 - index.jsp
 - listusers.jsp
 - ▶ resources
 - ▲ WEB-INF
 - classes
 - struts-config.xml
 - ▶ web.xml
 - ▶ test
 - ▶ target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Ibatis -->
<dependency>
    <groupId>org.apache.ibatis</groupId>
    <artifactId>ibatis-sqlmap</artifactId>
    <version>2.3.4.726</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
    <display-name> Struts1 iBatis Integration </display-name>

    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
            <param-value>/WEB-INF/struts-config.xml</param-value>
        </init-param>
```

```
<load-on-startup>2</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>

<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
  "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
  "http://struts.apache.org/dtds/struts-config_1_3.dtd">

<struts-config>

  <form-beans>
    <form-bean name="user" type="com.tutorials4u.model.User" />
  </form-beans>

  <action-mappings>
    <action path="/helloWorld"
      type="org.apache.struts.actions.ForwardAction"
      parameter="/index.jsp">
    </action>

    <action path="/insertUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/index.jsp" parameter="insertUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/listUsers" type="com.tutorials4u.actions.UserAction"
      parameter="listUsers">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/deleteUser" type="com.tutorials4u.actions.UserAction"
      parameter="deleteUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser" type="com.tutorials4u.actions.UserAction"
      parameter="editUser">
      <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/edituser.jsp" parameter="updateUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

  </action-mappings>
</struts-config>
```

Sql-maps-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE sqlMapConfig
    PUBLIC "-//ibatis.apache.org//DTD SQL Map Config 2.0//EN"
    "http://ibatis.apache.org/dtd/sql-map-config-2.dtd">
<sqlMapConfig>

    <settings useStatementNamespaces="true"/>

    <transactionManager type="JDBC">
        <dataSource type="SIMPLE">
            <property name="JDBC.Driver" value="com.mysql.jdbc.Driver"/>
            <property name="JDBC.ConnectionURL"
                value="jdbc:mysql://localhost:3306/hibernate"/>
            <property name="JDBC.Username" value="root"/>
            <property name="JDBC.Password" value=""/>
        </dataSource>
    </transactionManager>

    <sqlMap resource="user.xml"/>

</sqlMapConfig>
```

User.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sqlMap PUBLIC "-//ibatis.apache.org//DTD SQL Map 2.0//EN"
    "http://ibatis.apache.org/dtd/sql-map-2.dtd">
<sqlMap namespace="user">

    <typeAlias alias="USER" type="com.tutorials4u.model.User" />

    <resultMap id="userResultMap" class="USER">
        <result property="user_id" column="USER_ID" />
        <result property="username" column="USERNAME" />
        <result property="password" column="PASSWORD" />
        <result property="firstName" column="FIRSTNAME" />
        <result property="lastName" column="LASTNAME" />
        <result property="phone" column="PHONE" />
        <result property="emailId" column="EMAILID" />
    </resultMap>

    <select id="listUsers" resultMap="userResultMap">
        SELECT * FROM MYUSER
    </select>

    <insert id="insertUser" parameterClass="USER">
        INSERT INTO MYUSER
        (USERNAME,PASSWORD,FIRSTNAME, LASTNAME,PHONE,EMAILID)

        VALUES(#username#,#password#,#firstName#,#lastName#,#phone#,#emailId#);
    </insert>

    <delete id="deleteUser" parameterClass="java.lang.Integer">
        DELETE FROM MYUSER WHERE USER_ID = #value#
    </delete>
```

```
<select id="getUserById" parameterClass="java.Lang.Integer"
        resultMap="userResultMap">
    SELECT * FROM MYUSER WHERE USER_ID = #value#
</select>

<update id="updateUser" parameterClass="USER">
    UPDATE MYUSER
    SET USERNAME = #username#,
    PASSWORD     = #password#,
    FIRSTNAME    = #firstName#,
    LASTNAME     = #lastName#,
    PHONE        = #phone#,
    EMAILID      = #emailId#
    WHERE USER_ID = #user_id#
</update>
</sqlMap>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
    />

        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
import javax.persistence.Transient;
import org.apache.struts.action.ActionForm;
public class User extends ActionForm{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }
}
```



```
public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserDao userDao = new UserDao();

    public ActionForward insertUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.insertUser()");
        String result = new UserDao().insertUser(user);
        logger.debug("1 Record Inserted Successfully " + result);
        return mapping.findForward("success");
    }
}
```

```
public ActionForward listUsers(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.listUsers()");
    List<User> users = userDao.listUsers();
    request.setAttribute("users", users);
    return mapping.findForward("success");
}

public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.deleteUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    Logger.debug("value="+list.get(1));
    userDao.deleteUser(Integer.parseInt(list.get(1)));
    Logger.debug("1 Record deleted Successfully");
    return mapping.findForward("success");
}

public ActionForward editUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.editUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    User user=userDao.editUser(Integer.parseInt(list.get(1)));
    request.setAttribute("user",user);
    return mapping.findForward("success");
}

public ActionForward updateUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    User user = (User) form;
    Logger.debug("UserAction.updateUser()");
    userDao.updateUser(user);
    Logger.debug("1 Record Updated Successfully");
    return mapping.findForward("success");
}
}
```

IbatisUtil.java:

```
package com.tutorials4u.ibatis;
import java.io.IOException;
import java.io.Reader;
import com.ibatis.common.resources.Resources;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.ibatis.sqlmap.client.SqlMapClientBuilder;
public class IbatisUtil {
    private SqlMapClient sqlmapClient;
    public IbatisUtil() {
    }
    public static SqlMapClient initateSqlMapClient() {
        Reader reader = null;
        try {
            reader = Resources.getResourceAsReader("sql-maps-config.xml");
            SqlMapClient
            sqlmapClient = SqlMapClientBuilder.buildSqlMapClient(reader);
            return sqlmapClient;
        } catch (IOException e) {
            throw new RuntimeException(e.getMessage());
        }
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.ibatis.IbatisUtil;
import com.tutorials4u.model.User;

public class UserDao {
    private static final Logger logger = Logger.getLogger(UserAction.class);
    SqlMapClient sqlmapClient = IbatisUtil.initateSqlMapClient();

    public String insertUser(User user) {
        logger.debug("UserDao.insertUser()");
        try {
            logger.debug("Inserting values....");
            sqlmapClient.insert("user.insertUser", user);
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            return "failure";
        }
        return "success";
    }

    @SuppressWarnings("unchecked")
    public List<User> listUsers() throws SQLException {
        logger.debug("UserDao.listUsers()");
        User rec = new User();
        List<User> user = (List<User>) sqlmapClient.queryForList(
            "user.listUsers", rec);
        return user;
    }
}
```

```
public void deleteUser(int id) {
    logger.debug("UserDao.deleteUser()");
    try {
        sqlmapClient.delete("user.deleteUser", id);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public User editUser(int id) throws SQLException {
    logger.debug("UserDao.editUser()");
    User user = getUser(id);
    return user;
}

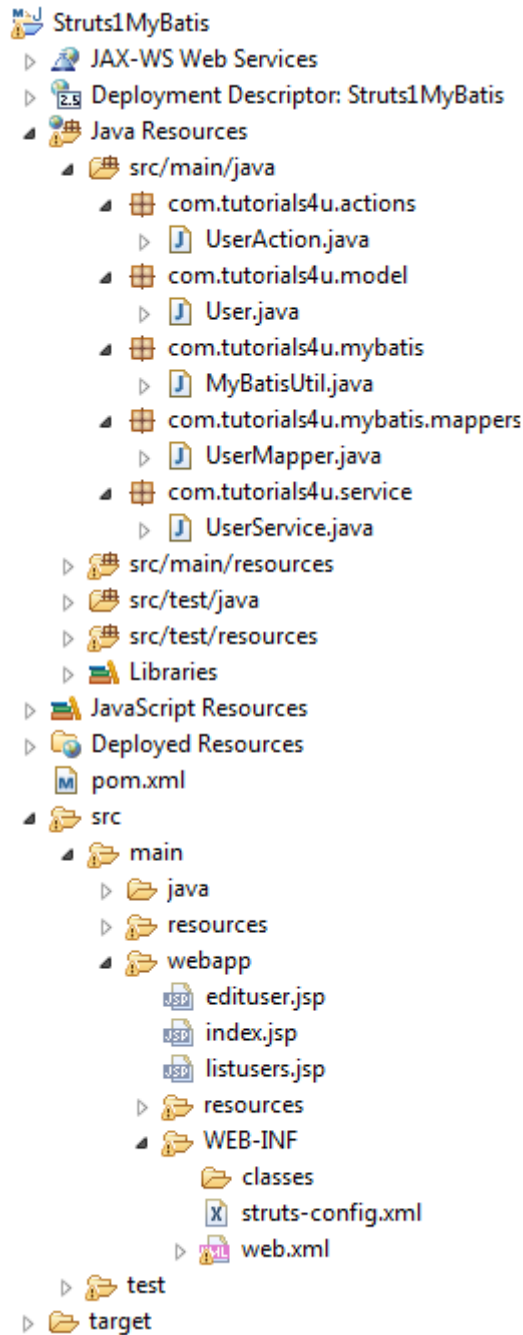
public User getUser(int id) throws SQLException {
    logger.debug("UserDao.getUser()");
    User user = (User) sqlmapClient.queryForObject("user.getUserById", id);
    return user;
}

public void updateUser(User user) {
    logger.debug("UserDao.updateUser()");
    try {
        sqlmapClient.update("user.updateUser", user);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}
```

Index.jsp, edituser.jsp, listUsers.jsp are same as above Struts1 JDBC Integration

➤ Struts1 MyBatis Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- MyBatis framework -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.1.1</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
    <display-name>Struts1 Mybatis Integration</display-name>

    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
```

```
<param-value>/WEB-INF/struts-config.xml</param-value>
</init-param>
<load-on-startup>2</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>

<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
  "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
  "http://struts.apache.org/dtds/struts-config_1_3.dtd">
<struts-config>
  <form-beans>
    <form-bean name="user" type="com.tutorials4u.model.User" />
  </form-beans>
  <action-mappings>
    <action path="/helloWorld"
type="org.apache.struts.actions.ForwardAction" parameter="/index.jsp">
    </action>

    <action path="/insertUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/index.jsp" parameter="insertUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/listUsers" type="com.tutorials4u.actions.UserAction"
      parameter="listUsers">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/deleteUser" type="com.tutorials4u.actions.UserAction"
      parameter="deleteUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser" type="com.tutorials4u.actions.UserAction"
      parameter="editUser">
      <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser" type="com.tutorials4u.actions.UserAction"
      name="user" input="/edituser.jsp" parameter="updateUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

  </action-mappings>
</struts-config>
```

mybatis-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
  <properties resource="jdbc.properties" />
  <typeAliases>
    <typeAlias type="com.tutorials4u.model.User"
      alias="User"></typeAlias>
  </typeAliases>
  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC" />
      <dataSource type="POOLED">
        <property name="driver" value="${jdbc.driverClassName}" />
        <property name="url" value="${jdbc.url}" />
        <property name="username" value="${jdbc.username}" />
        <property name="password" value="${jdbc.password}" />
      </dataSource>
    </environment>
  </environments>
  <mappers>
    <mapper resource="UserMapper.xml" />
  </mappers>
</configuration>
```

Jdbc.properties:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

UserMapper.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.tutorials4u.mybatis.mappers.UserMapper">

  <resultMap id="UserResultMap" type="User">
    <id property="user_id" column="USER_ID" />
    <result property="username" column="USERNAME"/>
    <result property="password" column="PASSWORD"/>
    <result property="firstName" column="FIRSTNAME"/>
    <result property="lastName" column="LASTNAME"/>
    <result property="phone" column="PHONE"/>
    <result property="emailId" column="EMAILID"/>
  </resultMap>
```



```
<insert id="insertUser" parameterType="user">
    INSERT INTO myuser
    (USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID)
    VALUES
    ({username},{password},{firstName},{lastName},{phone},{emailId
    })
</insert>

<select id="getAllUsers" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,FIRSTNAME,LASTNAME,PHONE,EMAILID
    FROM myuser;
</select>

<select id="getUserById" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
    FROM myuser
    WHERE USER_ID=#{user_id}
</select>

<update id="updateUser" parameterType="user">
    UPDATE myuser
    SET
    USERNAME = #{username},
    PASSWORD = #{password},
    FIRSTNAME= #{firstName},
    LASTNAME = #{lastName},
    PHONE    = #{phone},
    EMAILID  = #{emailId}
    WHERE
    USER_ID = #{user_id}
</update>

<delete id="deleteUser" parameterType="String">
    DELETE FROM
    myuser
    WHERE USER_ID = #{user_id}
</delete>

<select id="getUserByUsername" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
    FROM myuser
    WHERE USERNAME = #{username}
</select>
</mapper>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>
```

```
<!-- file -->
<appender name="file" class="org.apache.log4j.RollingFileAppender">
  <param name="append" value="false" />
  <param name="maxFileSize" value="10KB" />
  <param name="maxBackupIndex" value="5" />
  <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
/>

  <layout class="org.apache.log4j.PatternLayout">
    <param name="ConversionPattern"
      value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
  </layout>
</appender>

<root>
  <level value="DEBUG" />
  <appender-ref ref="console" />
  <appender-ref ref="file" />
</root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
import org.apache.struts.action.ActionForm;

public class User extends ActionForm {

  private static final long serialVersionUID = 1L;
  private int user_id;
  String username;
  String password;
  String firstName;
  String lastName;
  String phone;
  String emailId;

  public int getUser_id() {
    return user_id;
  }

  public void setUser_id(int user_id) {
    this.user_id = user_id;
  }

  public String getUsername() {
    return username;
  }

  public void setUsername(String username) {
    this.username = username;
  }

  public String getPassword() {
    return password;
  }
}
```

```
public void setPassword(String password) {
    this.password = password;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;
import com.tutorials4u.service.UserService;
```

```
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserService userDao = new UserService();

    public ActionForward insertUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.insertUser()");
        new UserService().insertUser(user);
        logger.debug("1 Record Inserted Successfully ");
        return mapping.findForward("success");
    }

    public ActionForward listUsers(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.listUsers()");
        List<User> users = userDao.getAllUsers();
        request.setAttribute("users", users);
        return mapping.findForward("success");
    }

    public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.deleteUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        logger.debug("value="+list.get(1));
        userDao.deleteUser(Integer.parseInt(list.get(1)));
        return mapping.findForward("success");
    }

    public ActionForward editUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.editUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        User user=userDao.getUserById(Integer.parseInt(list.get(1)));
        request.setAttribute("user",user);
        return mapping.findForward("success");
    }

    public ActionForward updateUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.updateUser()");
```

```
        userDao.updateUser(user);
        logger.debug("1 Record Updated Successfully");
        return mapping.findForward("success");
    }
}
```

UserMapper.java:

```
package com.tutorials4u.mybatis.mappers;
import java.util.List;
import com.tutorials4u.model.User;

public interface UserMapper {

    public void insertUser(User user);

    public User getUserById(Integer userId);

    public List<User> getAllUsers();

    public void updateUser(User user);

    public void deleteUser(Integer userId);
}
```

MyBatisUtil.java:

```
package com.tutorials4u.mybatis;
import java.io.IOException;
import java.io.Reader;
import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

public class MyBatisUtil {
    private static SqlSessionFactory factory;

    private MyBatisUtil() {
    }

    static {
        Reader reader = null;
        try {
            reader = Resources.getResourceAsReader("mybatis-config.xml");
        } catch (IOException e) {
            throw new RuntimeException(e.getMessage());
        }
        factory = new SqlSessionFactoryBuilder().build(reader);
    }

    public static SqlSessionFactory getSqlSessionFactory() {
        return factory;
    }
}
```

UserService.java:

```
package com.tutorials4u.service;
import java.util.List;
import org.apache.ibatis.session.SqlSession;
import com.tutorials4u.model.User;
import com.tutorials4u.mybatis.MyBatisUtil;
import com.tutorials4u.mybatis.mappers.UserMapper;

public class UserService {

    public void insertUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.insertUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }

    public User getUserById(Integer userId) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            return userMapper.getUserById(userId);
        } finally {
            sqlSession.close();
        }
    }

    public List<User> getAllUsers() {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            return userMapper.getAllUsers();
        } finally {
            sqlSession.close();
        }
    }

    public void updateUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.updateUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }
}
```

```
    }

    public void deleteUser(Integer userId) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.deleteUser(userId);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }
}
```

Index.jsp, edituser.jsp, listUsers.jsp are same as above Struts1 JDBC Integration

➤ Struts1 Spring JDBC Integration

Project Structure:

- └─ Struts1SpringJDBC
 - └─ JAX-WS Web Services
 - └─ Deployment Descriptor: Struts1SpringJDBC
 - └─ Loading descriptor for Struts1SpringJDBC...
 - └─ Java Resources
 - └─ src/main/java
 - └─ com.tutorials4u.actions
 - └─ UserAction.java
 - └─ com.tutorials4u.business
 - └─ BusinessDelegate.java
 - └─ com.tutorials4u.dao
 - └─ UserDao.java
 - └─ com.tutorials4u.jdbc
 - └─ JDBCUtil.java
 - └─ com.tutorials4u.model
 - └─ User.java
 - └─ src/main/resources
 - └─ META-INF
 - └─ jdbc.properties
 - └─ log4j.xml
 - └─ src/test/java
 - └─ src/test/resources
 - └─ Libraries
 - └─ JavaScript Resources
 - └─ Deployed Resources
 - └─ pom.xml
 - └─ src
 - └─ main
 - └─ java
 - └─ resources
 - └─ webapp
 - └─ edituser.jsp
 - └─ index.jsp
 - └─ listusers.jsp
 - └─ resources
 - └─ WEB-INF
 - └─ ApplicationContext.xml
 - └─ classes
 - └─ struts-config.xml
 - └─ web.xml
 - └─ test
 - └─ target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Struts + Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-struts</artifactId>
    <version>2.0.8</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
    <display-name>Struts1 Spring JDBC Integration </display-name>

    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
            <param-value>/WEB-INF/struts-config.xml</param-value>
```

```
</init-param>
<load-on-startup>2</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>

<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
  "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
  "http://struts.apache.org/dtds/struts-config_1_3.dtd">
<struts-config>
  <form-beans>
    <form-bean name="user" type="com.tutorials4u.model.User" />
  </form-beans>

  <action-mappings>

    <action path="/insertUser"
      type="org.springframework.web.struts.DelegatingActionProxy"
      name="user"
      input="/index.jsp" parameter="insertUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/listUsers"
      type="org.springframework.web.struts.DelegatingActionProxy"
      parameter="listUsers">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/deleteUser"
      type="org.springframework.web.struts.DelegatingActionProxy"
      parameter="deleteUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser"
      type="org.springframework.web.struts.DelegatingActionProxy"
      parameter="editUser">
      <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser"
      type="org.springframework.web.struts.DelegatingActionProxy"
      name="user"
      input="/edituser.jsp" parameter="updateUser">
      <forward name="success" path="/listusers.jsp" />
    </action>
  </action-mappings>
```

```
<plug-in className="org.springframework.web.struts.ContextLoaderPlugIn">
    <set-property property="contextConfigLocation" value="/WEB-INF/ApplicationContext.xml" />
</plug-in>
</struts-config>
```

ApplicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <!-- Business Objects -->
    <bean id="businessDelegate"
        class="com.tutorials4u.business.BusinessDelegate">
    </bean>

    <!-- Actions Classes -->
    <bean name="/insertUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/listUsers" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/deleteUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/editUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/updateUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>
</beans>
```

Jdbc.properties:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
```

```
        value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
</appender>

<!-- file -->
<appender name="file" class="org.apache.log4j.RollingFileAppender">
    <param name="append" value="false" />
    <param name="maxFileSize" value="10KB" />
    <param name="maxBackupIndex" value="5" />
    <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.log"
/>
    <layout class="org.apache.log4j.PatternLayout">
        <param name="ConversionPattern"
            value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
</appender>

<root>
    <level value="DEBUG" />
    <appender-ref ref="console" />
    <appender-ref ref="file" />
</root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
import javax.persistence.Transient;
import org.apache.struts.action.ActionForm;
```

```
public class User extends ActionForm{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }
}
```

```
public String getUsername() {
    return username;
}

public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;

import com.tutorials4u.business.BusinessDelegate;
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    // Get a logger
    private static final Logger logger = Logger
        .getLogger(UserAction.class);
    BusinessDelegate businessDelegate;

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }

    public void setBusinessDelegate(BusinessDelegate businessDeletage) {
        this.businessDelegate = businessDeletage;
    }

    public ActionForward listUsers(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.listUsers()");
        List<User> users=this.businessDelegate.listUsers();
        request.setAttribute("users", users);
        return mapping.findForward("success");
    }

    public ActionForward insertUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.insertUser()");
        String result=this.businessDelegate.insertUser(user);
        logger.debug("1 Record Inserted Successfully " + result);
        return mapping.findForward("success");
    }

    public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.deleteUser()");
        String userid = request.getQueryString();
        ArrayList<String> list = new ArrayList<String>();
        for (String newstr : userid.split("=")) {
            list.add(newstr);
        }
    }
}
```

```
        logger.debug("value=" + list.get(1));
        this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
        return mapping.findForward("success");
    }

    public ActionForward editUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.editUser()");
        String userid = request.getQueryString();
        ArrayList<String> list = new ArrayList<String>();
        for (String newstr : userid.split("=")) {
            list.add(newstr);
        }
        User user =
            this.businessDelegate.editUser(Integer.parseInt(list.get(1)));
        request.setAttribute("user", user);
        return mapping.findForward("success");
    }

    public ActionForward updateUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.updateUser()");
        this.businessDelegate.updateUser(user);
        logger.debug("1 Record Updated Successfully");
        return mapping.findForward("success");
    }
}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;
import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserDao userDao = new UserDao();

    public String insertUser(User user) throws IOException, SQLException{
        logger.debug("BusinessDelegate.insertUser()");
        String result = new UserDao().insertUser(user);
        return result;
    }

    public List<User> listUsers() throws IOException, SQLException {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.listUsers();
        return users;
    }
}
```

```
public void deleteUser(int userid) throws IOException, SQLException {
    logger.debug("BusinessDelegate.deleteUser()");
    userDao.deleteUser(userid);
}

public User editUser(int userid) throws IOException, SQLException{
    logger.debug("UserAction.editUser()");
    User user = userDao.editUser(userid);
    return user;
}

public void updateUser(User user) throws IOException {
    logger.debug("UserAction.updateUser()");
    userDao.updateUser(user);
}
}
```

UserDao.java:

```
package com.tutorials4u.dao;
import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.jdbc.JDBCUtil;
import com.tutorials4u.model.User;

public class UserDao {
    private static final Logger logger = Logger.getLogger(UserAction.class);

    PreparedStatement stmt = null;

    @SuppressWarnings("static-access")
    public String insertUser(User newuser) throws IOException, SQLException {
        logger.debug("UserDao.createUser()");
        JDBCUtil util = new JDBCUtil();
        Connection conn = util.getMysqlConnection();
        try {
            String username = newuser.getUsername();
            String password = newuser.getPassword();
            String firstname = newuser.getFirstName();
            String lastname = newuser.getLastName();
            String emailId = newuser.getEmailId();
            String phone = newuser.getPhone();

            String sql = new String(
                "insert into
myuser(USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID) values(?,?,?,?,?,?)");

            stmt = conn.prepareStatement(sql);
            stmt.setString(1, username);

```



```
        stmt.setString(2, password);
        stmt.setString(3, firstname);
        stmt.setString(4, lastname);
        stmt.setString(5, phone);
        stmt.setString(6, emailId);

        logger.debug("Inserting values....");
        stmt.executeUpdate();
    } catch (Exception e) {
        logger.error("Error creating User :" + e.getMessage());
        return "failure";
    } finally {
        stmt.close();
        conn.close();
    }
    return "success";
}

@SuppressWarnings("static-access")
public List<User> listUsers() throws IOException, SQLException {
    logger.debug("UserDao.listUsers()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    List<User> users = new ArrayList<User>();
    try {
        String sql = "select
USER_ID,USERNAME,FIRSTNAME,LASTNAME,PHONE,EMAILID from myuser";
        statement = conn.createStatement();
        result = statement.executeQuery(sql);
        User newuser = null;
        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
            users.add(newuser);
        }
    } catch (SQLException e) {
        logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
        // conn.close();
    }
    return users;
}

@SuppressWarnings("static-access")
public void deleteUser(int id) throws IOException, SQLException {
    logger.debug("UserDao.deleteUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    Statement statement = null;
    try {
```

```
        String sql = "delete from myuser where user_id=" + id;
        statement = conn.createStatement();
        statement.executeUpdate(sql);
    } catch (SQLException e) {
        logger.error("Error deleting User :" + e.getMessage());
    } finally {
        statement.close();
        // conn.close();
    }
}

@SuppressWarnings("static-access")
public User editUser(int id) throws IOException, SQLException {
    logger.debug("UserDao.editUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    User user = getUser(id);
    return user;
}

@SuppressWarnings("static-access")
public User getUser(int id) throws SQLException, IOException {
    logger.debug("UserDao.getUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    User newuser = null;
    try {
        String sql = "select
USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID from myuser WHERE
USER_ID="
                + id;
        statement = conn.createStatement();
        result = statement.executeQuery(sql);

        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setPassword(result.getString("PASSWORD"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
        }
    } catch (SQLException e) {
        logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
        // conn.close();
    }
    return newuser;
}

@SuppressWarnings("static-access")
public void updateUser(User newuser) throws IOException {
    logger.debug("UserDao.updateUser()");
}
```

```
JDBCUtil util = new JDBCUtil();
Connection conn = util.getMysqlConnection();
try {
    int userid = newuser.getUser_id();
    String username = newuser.getUsername();
    String password = newuser.getPassword();
    String firstname = newuser.getFirstName();
    String lastname = newuser.getLastName();
    String emailId = newuser.getEmailId();
    String phone = newuser.getPhone();

    String sql = "UPDATE MYUSER SET
USERNAME=?, PASSWORD=?, FIRSTNAME=?, LASTNAME=?, PHONE=?, EMAILID=? WHERE
USER_ID="+userid;

    stmt = conn.prepareStatement(sql);
    stmt.setString(1, username);
    stmt.setString(2, password);
    stmt.setString(3, firstname);
    stmt.setString(4, lastname);
    stmt.setString(5, phone);
    stmt.setString(6, emailId);

    logger.debug("Updating values....");
    stmt.executeUpdate();
} catch (Exception e) {
    logger.error("Error updating User :" + e.getMessage());
}
}
```

Index.jsp, edituser.jsp, listUsers.jsp are same as above Struts1 JDBC Integration

➤ Struts1 Spring Hibernate Integration

Project Structure:

- └─ Struts1SpringHibernate
 - └─ JAX-WS Web Services
 - └─ Deployment Descriptor: Struts1SpringHibernate
 - └─ Java Resources
 - └─ src/main/java
 - └─ com.tutorials4u.actions
 - └─ UserAction.java
 - └─ com.tutorials4u.business
 - └─ BusinessDelegate.java
 - └─ com.tutorials4u.dao
 - └─ UserDao.java
 - └─ com.tutorials4u.hibernate
 - └─ HibernateUtil.java
 - └─ HibernateUtil
 - └─ com.tutorials4u.model
 - └─ User.java
 - └─ src/main/resources
 - └─ META-INF
 - └─ hibernate.cfg.xml
 - └─ log4j.xml
 - └─ User.hbm.xml
 - └─ src/test/java
 - └─ src/test/resources
 - └─ Libraries
 - └─ JavaScript Resources
 - └─ Deployed Resources
 - └─ pom.xml
 - └─ src
 - └─ main
 - └─ java
 - └─ resources
 - └─ webapp
 - └─ edituser.jsp
 - └─ index.jsp
 - └─ listusers.jsp
 - └─ resources
 - └─ WEB-INF
 - └─ ApplicationContext.xml
 - └─ classes
 - └─ struts-config.xml
 - └─ web.xml
 - └─ test
 - └─ target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Hibernate framework -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>4.2.4.Final</version>
</dependency>

<!-- Struts + Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-struts</artifactId>
    <version>2.0.8</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
    <display-name>Struts1 Spring Hibernate Integration </display-name>

    <servlet>
        <servlet-name>action</servlet-name>
        <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
        <init-param>
            <param-name>config</param-name>
            <param-value>/WEB-INF/struts-config.xml</param-value>
        </init-param>
        <load-on-startup>2</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>action</servlet-name>
        <url-pattern>*.do</url-pattern>
    </servlet-mapping>

    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
    "http://struts.apache.org/dtds/struts-config_1_3.dtd">
<struts-config>
    <form-beans>
        <form-bean name="user" type="com.tutorials4u.model.User" />
    </form-beans>

    <action-mappings>

        <action path="/insertUser"
            type="org.springframework.web.struts.DelegatingActionProxy"
            name="user"
            input="/index.jsp" parameter="insertUser">
            <forward name="success" path="/listusers.jsp" />
        </action>

        <action path="/listUsers"
            type="org.springframework.web.struts.DelegatingActionProxy"
            parameter="listUsers">
            <forward name="success" path="/listusers.jsp" />
        </action>

        <action path="/deleteUser"
```

```
        type="org.springframework.web.struts.DelegatingActionProxy"
        parameter="deleteUser">
        <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser"
        type="org.springframework.web.struts.DelegatingActionProxy"
        parameter="editUser">
        <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser"
        type="org.springframework.web.struts.DelegatingActionProxy"
name="user"
        input="/edituser.jsp" parameter="updateUser">
        <forward name="success" path="/listusers.jsp" />
    </action>
</action-mappings>

    <plug-in className="org.springframework.web.struts.ContextLoaderPlugIn">
        <set-property property="contextConfigLocation" value="/WEB-INF/ApplicationContext.xml" />
    </plug-in>
</struts-config>
```

ApplicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <!-- Business Objects -->
    <bean id="businessDelegate"
        class="com.tutorials4u.business.BusinessDelegate">
    </bean>

    <!-- Actions Classes -->
    <bean name="/insertUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/listUsers" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/deleteUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/editUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/updateUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>
</beans>
```

Hibernate.cfg.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property
            name="hibernate.connection.driver_class">com.mysql.jdbc.Driver
        </property>
        <property
            name="hibernate.connection.url">jdbc:mysql://localhost:3306/mydb
        </property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password"></property>
        <property
            name="hibernate.dialect">org.hibernate.dialect.MySQLDialect
        </property>
        <property name="show_sql">>true</property>
        <property name="format_sql">>true</property>
        <property name="hbm2ddl.auto">update</property>
        <mapping resource="User.hbm.xml" />
    </session-factory>
</hibernate-configuration>
```

User.hbm.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.tutorials4u.model.User" table="MYUSER">
        <meta attribute="class-description">
            This class contains the User detail.
        </meta>
        <id name="user_id" type="int" column="USER_ID">
            <generator class="native"/>
        </id>
        <property name="username" column="USERNAME" type="string"/>
        <property name="password" column="PASSWORD" type="string"/>
        <property name="firstName" column="FIRSTNAME" type="string"/>
        <property name="lastName" column="LASTNAME" type="string"/>
        <property name="phone" column="PHONE" type="string"/>
        <property name="emailId" column="EMAILID" type="string"/>
    </class>
</hibernate-mapping>
```


Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
    />
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
import javax.persistence.Transient;
import org.apache.struts.action.ActionForm;

public class User extends ActionForm{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
```

```
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }
}
```

```
@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;

import com.tutorials4u.business.BusinessDelegate;
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    // Get a logger
    private static final Logger logger = Logger
        .getLogger(UserAction.class);
    BusinessDelegate businessDelegate;

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }

    public void setBusinessDelegate(BusinessDelegate businessDeletage) {
        this.businessDelegate = businessDeletage;
    }

    public ActionForward listUsers(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.listUsers()");
        List<User> users=this.businessDelegate.listUsers();
        request.setAttribute("users", users);
        return mapping.findForward("success");
    }

    public ActionForward insertUser(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        User user = (User) form;
        logger.debug("UserAction.insertUser()");
        String result=this.businessDelegate.insertUser(user);
        logger.debug("1 Record Inserted Successfully " + result);
        return mapping.findForward("success");
    }
}
```

```
public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    logger.debug("UserAction.deleteUser()");
    String userid = request.getQueryString();
    ArrayList<String> list = new ArrayList<String>();
    for (String newstr : userid.split("=")) {
        list.add(newstr);
    }
    logger.debug("value=" + list.get(1));
    this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
    return mapping.findForward("success");
}

public ActionForward editUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    logger.debug("UserAction.editUser()");
    String userid = request.getQueryString();
    ArrayList<String> list = new ArrayList<String>();
    for (String newstr : userid.split("=")) {
        list.add(newstr);
    }
    User user =
this.businessDelegate.editUser(Integer.parseInt(list.get(1)));
    request.setAttribute("user", user);
    return mapping.findForward("success");
}

public ActionForward updateUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    User user = (User) form;
    logger.debug("UserAction.updateUser()");
    this.businessDelegate.updateUser(user);
    logger.debug("1 Record Updated Successfully");
    return mapping.findForward("success");
}
}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;

import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserDao userDao = new UserDao();

    public String insertUser(User user){
        logger.debug("BusinessDelegate.insertUser()");
        String result = new UserDao().insertUser(user);
    }
}
```

```
        return result;
    }

    public List<User> listUsers() {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.listUsers();
        return users;
    }

    public void deleteUser(int userid) {
        logger.debug("BusinessDelegate.deleteUser()");
        userDao.deleteUser(userid);
    }

    public User editUser(int userid){
        logger.debug("UserAction.editUser()");
        User user = userDao.editUser(userid);
        return user;
    }

    public void updateUser(User user) {
        logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
    }
}
```

HibernateUtil.java:

```
package com.tutorials4u.hibernate;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class HibernateUtil {

    private static final SessionFactory sessionFactory =
        buildSessionFactory();

    private static SessionFactory buildSessionFactory() {
        try {
            // Create the SessionFactory from hibernate.cfg.xml
            return new Configuration().configure().buildSessionFactory();
        }
        catch (Throwable ex) {
            // Make sure you log the exception, as it might be swallowed
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

    public static void shutdown() {
        // Close caches and connection pools
        getSessionFactory().close();
    }
}
```

```
}
```

UserDao.java:

```
package com.tutorials4u.dao;

import java.util.List;
import org.hibernate.Session;
import org.apache.log4j.Logger;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.hibernate.HibernateUtil;
import com.tutorials4u.model.User;

public class UserDao {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    Session session = HibernateUtil.getSessionFactory().openSession();

    public String insertUser(User newUser) {
        logger.debug("UserDao.insertUser()");
        try {
            session.beginTransaction();
            User user = new User();
            user.setUsername(newUser.getUsername());
            user.setPassword(newUser.getPassword());
            user.setFirstName(newUser.getFirstName());
            user.setLastName(newUser.getLastName());
            user.setEmailId(newUser.getEmailId());
            user.setPhone(newUser.getPhone());
            logger.debug("Inserting values....");
            session.save(user);
            session.getTransaction().commit();
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            session.getTransaction().rollback();
            return "failure";
        }
        return "success";
    }

    @SuppressWarnings("unchecked")
    public List<User> listUsers() {
        logger.debug("UserDao.listUsers()");
        return session.createQuery("From User").list();
    }

    public void deleteUser(int id) {
        logger.debug("UserDao.deleteUser()");
        session.beginTransaction();
        User user = getUser(id);
        session.delete(user);
        session.getTransaction().commit();
    }

    public User editUser(int id) {
        logger.debug("UserDao.editUser()");
        User user = getUser(id);
        return user;
    }
}
```

```
public User getUser(int id) {
    logger.debug("UserDao.getUser()");
    User user = (User) session.get(User.class, id);
    return user;
}

public void updateUser(User user) {
    logger.debug("UserDao.updateUser()");
    session.beginTransaction();
    session.update(user);
    session.getTransaction().commit();
}
}
```

Index.jsp, edituser.jsp, listUsers.jsp are same as above Struts1 JDBC Integration

➤ Struts1 Spring iBatis Integration

Project Structure:

- ▲ Struts1SpringIbatis
 - ▶ JAX-WS Web Services
 - ▶ Deployment Descriptor: Struts1SpringIbatis
 - ▲ Java Resources
 - ▲ src/main/java
 - ▲ com.tutorials4u.actions
 - ▶ UserAction.java
 - ▲ com.tutorials4u.business
 - ▶ BusinessDelegate.java
 - ▲ com.tutorials4u.dao
 - ▶ UserDao.java
 - ▲ com.tutorials4u.ibatis
 - ▶ IbatisUtil.java
 - ▲ com.tutorials4u.model
 - ▶ User.java
 - ▲ src/main/resources
 - META-INF
 - log4j.xml
 - sql-maps-config.xml
 - user.xml
 - ▶ src/test/java
 - ▶ src/test/resources
 - ▶ Libraries
 - ▶ JavaScript Resources
 - ▶ Deployed Resources
 - pom.xml
- ▲ src
 - ▲ main
 - ▶ java
 - ▶ resources
 - ▲ webapp
 - edituser.jsp
 - index.jsp
 - listusers.jsp
 - ▶ resources
 - ▲ WEB-INF
 - ApplicationContext.xml
 - classes
 - struts-config.xml
 - ▶ web.xml
 - ▶ test
 - ▶ target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Ibatis framework -->
<dependency>
    <groupId>org.apache.ibatis</groupId>
    <artifactId>ibatis-sqlmap</artifactId>
    <version>2.3.4.726</version>
</dependency>

<!-- Struts + Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-struts</artifactId>
    <version>2.0.8</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
  <display-name>Struts1 Spring iBatis Integration </display-name>

  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <init-param>
      <param-name>config</param-name>
      <param-value>/WEB-INF/struts-config.xml</param-value>
    </init-param>
    <load-on-startup>2</load-on-startup>
  </servlet>

  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>

  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
  "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
  "http://struts.apache.org/dtds/struts-config_1_3.dtd">
<struts-config>
  <form-beans>
    <form-bean name="user" type="com.tutorials4u.model.User" />
  </form-beans>

  <action-mappings>

    <action path="/insertUser"
      type="org.springframework.web.struts.DelegatingActionProxy"
      name="user"
      input="/index.jsp" parameter="insertUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/listUsers"
      type="org.springframework.web.struts.DelegatingActionProxy"
      parameter="listUsers">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/deleteUser"
```

```
        type="org.springframework.web.struts.DelegatingActionProxy"
        parameter="deleteUser">
        <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser"
        type="org.springframework.web.struts.DelegatingActionProxy"
        parameter="editUser">
        <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser"
        type="org.springframework.web.struts.DelegatingActionProxy"
        name="user"
        input="/edituser.jsp" parameter="updateUser">
        <forward name="success" path="/listusers.jsp" />
    </action>
</action-mappings>

<plug-in className="org.springframework.web.struts.ContextLoaderPlugIn">
    <set-property property="contextConfigLocation" value="/WEB-INF/ApplicationContext.xml" />
</plug-in>
</struts-config>
```

ApplicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <!-- Business Objects -->
    <bean id="businessDelegate"
        class="com.tutorials4u.business.BusinessDelegate">
    </bean>

    <!-- Actions Classes -->
    <bean name="/insertUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/listUsers" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/deleteUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/editUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/updateUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>
</beans>
```

sql-maps-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE sqlMapConfig
    PUBLIC "-//ibatis.apache.org//DTD SQL Map Config 2.0//EN"
    "http://ibatis.apache.org/dtd/sql-map-config-2.dtd">
<sqlMapConfig>

    <settings useStatementNamespaces="true"/>

    <transactionManager type="JDBC">
        <dataSource type="SIMPLE">
            <property name="JDBC.Driver" value="com.mysql.jdbc.Driver"/>
            <property name="JDBC.ConnectionURL"
                value="jdbc:mysql://localhost:3306/hibernate"/>
            <property name="JDBC.Username" value="root"/>
            <property name="JDBC.Password" value=""/>
        </dataSource>
    </transactionManager>

    <sqlMap resource="user.xml"/>

</sqlMapConfig>
```

User.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sqlMap PUBLIC "-//ibatis.apache.org//DTD SQL Map 2.0//EN"
    "http://ibatis.apache.org/dtd/sql-map-2.dtd">
<sqlMap namespace="user">

    <typeAlias alias="USER" type="com.tutorials4u.model.User" />

    <resultMap id="userResultMap" class="USER">
        <result property="user_id" column="USER_ID" />
        <result property="username" column="USERNAME" />
        <result property="password" column="PASSWORD" />
        <result property="firstName" column="FIRSTNAME" />
        <result property="lastName" column="LASTNAME" />
        <result property="phone" column="PHONE" />
        <result property="emailId" column="EMAILID" />
    </resultMap>

    <select id="listUsers" resultMap="userResultMap">
        SELECT * FROM MYUSER
    </select>

    <insert id="insertUser" parameterClass="USER">
        INSERT INTO MYUSER
        (USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID)

        VALUES(#username#,#password#,#firstName#,#lastName#,#phone#,#emailId#);
    </insert>

    <delete id="deleteUser" parameterClass="java.Lang.Integer">
        DELETE FROM MYUSER WHERE USER_ID = #value#
    </delete>
```

```
</delete>

<select id="getUserById" parameterClass="java.Lang.Integer"
        resultMap="userResultMap">
    SELECT * FROM MYUSER WHERE USER_ID = #value#
</select>

<update id="updateUser" parameterClass="USER">
    UPDATE MYUSER
    SET USERNAME = #username#,
    PASSWORD     = #password#,
    FIRSTNAME    = #firstName#,
    LASTNAME     = #lastName#,
    PHONE        = #phone#,
    EMAILID      = #emailId#
    WHERE USER_ID = #user_id#
</update>

</sqlMap>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
    />

    <layout class="org.apache.Log4j.PatternLayout">
        <param name="ConversionPattern"
            value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
</appender>

<root>
    <level value="DEBUG" />
    <appender-ref ref="console" />
    <appender-ref ref="file" />
</root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
import javax.persistence.Transient;
import org.apache.struts.action.ActionForm;

public class User extends ActionForm{

    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
}
```

```
public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;
import com.tutorials4u.business.BusinessDelegate;
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    // Get a logger
    private static final Logger logger = Logger
        .getLogger(UserAction.class);
    BusinessDelegate businessDelegate;

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }
}
```

```
public void setBusinessDelegate(BusinessDelegate businessDeletage) {
    this.businessDelegate = businessDeletage;
}

public ActionForward listUsers(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.listUsers()");
    List<User> users=this.businessDelegate.listUsers();
    request.setAttribute("users", users);
    return mapping.findForward("success");
}

public ActionForward insertUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    User user = (User) form;
    Logger.debug("UserAction.insertUser()");
    String result=this.businessDelegate.insertUser(user);
    Logger.debug("1 Record Inserted Successfully " + result);
    return mapping.findForward("success");
}

public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.deleteUser()");
    String userid = request.getQueryString();
    ArrayList<String> list = new ArrayList<String>();
    for (String newstr : userid.split("=")) {
        list.add(newstr);
    }
    Logger.debug("value=" + list.get(1));
    this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
    return mapping.findForward("success");
}

public ActionForward editUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.editUser()");
    String userid = request.getQueryString();
    ArrayList<String> list = new ArrayList<String>();
    for (String newstr : userid.split("=")) {
        list.add(newstr);
    }
    User user =
    this.businessDelegate.editUser(Integer.parseInt(list.get(1)));
    request.setAttribute("user", user);
    return mapping.findForward("success");
}
```



```
public ActionForward updateUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    User user = (User) form;
    logger.debug("UserAction.updateUser()");
    this.businessDelegate.updateUser(user);
    logger.debug("1 Record Updated Successfully");
    return mapping.findForward("success");
}

}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserDao userDao = new UserDao();

    public String insertUser(User user){
        logger.debug("BusinessDelegate.insertUser()");
        String result = new UserDao().insertUser(user);
        return result;
    }

    public List<User> listUsers() throws SQLException {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.listUsers();
        return users;
    }

    public void deleteUser(int userid) {
        logger.debug("BusinessDelegate.deleteUser()");
        userDao.deleteUser(userid);
    }

    public User editUser(int userid) throws SQLException{
        logger.debug("UserAction.editUser()");
        User user = userDao.editUser(userid);
        return user;
    }

    public void updateUser(User user) {
        logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
    }

}
```

IbatisUtil.java:

```
package com.tutorials4u.ibatis;

import java.io.IOException;
import java.io.Reader;
import com.ibatis.common.resources.Resources;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.ibatis.sqlmap.client.SqlMapClientBuilder;

public class IbatisUtil {
    private SqlMapClient sqlmapClient;

    public IbatisUtil() {
    }

    public static SqlMapClient initateSqlMapClient() {
        Reader reader = null;
        try {
            reader = Resources.getResourceAsReader("sql-maps-config.xml");
            SqlMapClient sqlmapClient = SqlMapClientBuilder
                .buildSqlMapClient(reader);
            return sqlmapClient;
        } catch (IOException e) {
            throw new RuntimeException(e.getMessage());
        }
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.ibatis.IbatisUtil;
import com.tutorials4u.model.User;

public class UserDao {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    SqlMapClient sqlmapClient = IbatisUtil.initateSqlMapClient();

    public String insertUser(User user) {
        logger.debug("UserDao.insertUser()");
        try {
            logger.debug("Inserting values....");
            sqlmapClient.insert("user.insertUser", user);
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            return "failure";
        }
        return "success";
    }
}
```

```
@SuppressWarnings("unchecked")
public List<User> listUsers() throws SQLException {
    logger.debug("UserDao.listUsers()");
    User rec = new User();
    List<User> user = (List<User>) sqlmapClient.queryForList(
        "user.listUsers", rec);
    return user;
}

public void deleteUser(int id) {
    logger.debug("UserDao.deleteUser()");
    try {
        sqlmapClient.delete("user.deleteUser", id);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public User editUser(int id) throws SQLException {
    logger.debug("UserDao.editUser()");
    User user = getUser(id);
    return user;
}
















































public User getUser(int id) throws SQLException {
    logger.debug("UserDao.getUser()");
    User user = (User) sqlmapClient.queryForObject("user.getUserById",
        id);
    return user;
}

public void updateUser(User user) {
    logger.debug("UserDao.updateUser()");
    try {
        sqlmapClient.update("user.updateUser", user);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}
```

Index.jsp, edituser.jsp, listUsers.jsp are same as above Struts1 JDBC Integration

➤ Struts1 Spring MyBatis Integration

Project Structure:

- ▲  Struts1SpringMyBatis
 - ▷  JAX-WS Web Services
 - ▷  Deployment Descriptor: Struts1SpringMyBatis
 - ▲  Java Resources
 - ▲  src/main/java
 - ▲  com.tutorials4u.actions
 - ▷  UserAction.java
 - ▲  com.tutorials4u.business
 - ▷  BusinessDelegate.java
 - ▲  com.tutorials4u.model
 - ▷  User.java
 - ▲  com.tutorials4u.mybatis
 - ▷  MyBatisUtil.java
 - ▲  com.tutorials4u.mybatis.mappers
 - ▷  UserMapper.java
 - ▲  com.tutorials4u.service
 - ▷  UserService.java
 - ▲  src/main/resources
 - ▲  META-INF
 - ▷  jdbc.properties
 - ▲  log4j.xml
 - ▲  mybatis-config.xml
 - ▲  UserMapper.xml
 - ▷  src/test/java
 - ▷  src/test/resources
 - ▷  Libraries
 - ▷  JavaScript Resources
 - ▷  Deployed Resources
 - ▲  pom.xml
- ▲  src
 - ▲  main
 - ▷  java
 - ▷  resources
 - ▲  webapp
 - ▲  edituser.jsp
 - ▲  index.jsp
 - ▲  listusers.jsp
 - ▷  resources
 - ▲  WEB-INF
 - ▲  ApplicationContext.xml
 - ▲  classes
 - ▲  jboss-deployment-structure.xml
 - ▷  lib
 - ▲  struts-config.xml
 - ▷  web.xml
 - ▷  test
 - ▷  target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-core</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-taglib</artifactId>
    <version>1.3.10</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts-extras</artifactId>
    <version>1.3.10</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Mybatis framework -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.1.1</version>
</dependency>

<!-- Struts + Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-struts</artifactId>
    <version>2.0.8</version>
</dependency>
```

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_2_5.xsd" id="WebApp_ID" version="2.5">
  <display-name> Struts1 Spring MyBatis Integration </display-name>

  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <init-param>
      <param-name>config</param-name>
      <param-value>/WEB-INF/struts-config.xml</param-value>
    </init-param>
    <load-on-startup>2</load-on-startup>
  </servlet>

  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>

  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

Struts-config.xml:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
"http://struts.apache.org/dtds/struts-config_1_3.dtd">
<struts-config>
  <form-beans>
    <form-bean name="user" type="com.tutorials4u.model.User" />
  </form-beans>

  <action-mappings>

    <action path="/insertUser"
      type="org.springframework.web.struts.DelegatingActionProxy"
      name="user"
      input="/index.jsp" parameter="insertUser">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/listUsers"
      type="org.springframework.web.struts.DelegatingActionProxy"
      parameter="listUsers">
      <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/deleteUser"
      type="org.springframework.web.struts.DelegatingActionProxy"
      parameter="deleteUser">
```

```
        <forward name="success" path="/listusers.jsp" />
    </action>

    <action path="/editUser"
        type="org.springframework.web.struts.DelegatingActionProxy"
        parameter="editUser">
        <forward name="success" path="/edituser.jsp" />
    </action>

    <action path="/updateUser"
        type="org.springframework.web.struts.DelegatingActionProxy"
        name="user"
        input="/edituser.jsp" parameter="updateUser">
        <forward name="success" path="/listusers.jsp" />
    </action>
</action-mappings>

    <plug-in className="org.springframework.web.struts.ContextLoaderPlugIn">
        <set-property property="contextConfigLocation" value="/WEB-INF/ApplicationContext.xml" />
    </plug-in>
</struts-config>
```

ApplicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
    "http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <!-- Business Objects -->
    <bean id="businessDelegate"
        class="com.tutorials4u.business.BusinessDelegate">
    </bean>

    <!-- Actions Classes -->
    <bean name="/insertUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/listUsers" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/deleteUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/editUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>

    <bean name="/updateUser" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref local="businessDelegate"/></property>
    </bean>
</beans>
```

mybatis-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
  <properties resource="jdbc.properties" />
  <typeAliases>
    <typeAlias type="com.tutorials4u.model.User"
      alias="User"></typeAlias>
  </typeAliases>
  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC" />
      <dataSource type="POOLED">
        <property name="driver" value="${jdbc.driverClassName}" />
        <property name="url" value="${jdbc.url}" />
        <property name="username" value="${jdbc.username}" />
        <property name="password" value="${jdbc.password}" />
      </dataSource>
    </environment>
  </environments>
  <mappers>
    <!-- <mapper
      resource="com/tutorials4u/mybatis/mappers/UserMapper.xml" /> -->
    <mapper resource="UserMapper.xml" />
  </mappers>
</configuration>
```

Jdbc.properties:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

UserMapper.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.tutorials4u.mybatis.mappers.UserMapper">

  <resultMap id="UserResultMap" type="User">
    <id property="user_id" column="USER_ID" />
    <result property="username" column="USERNAME"/>
    <result property="password" column="PASSWORD"/>
    <result property="firstName" column="FIRSTNAME"/>
    <result property="lastName" column="LASTNAME"/>
    <result property="phone" column="PHONE"/>
    <result property="emailId" column="EMAILID"/>
  </resultMap>
```



```
<insert id="insertUser" parameterType="user">
    INSERT INTO myuser
    (USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID)
    VALUES
    ({username},{password},{firstName},{lastName},{phone},{emailId
    })
</insert>

<select id="getAllUsers" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,FIRSTNAME,LASTNAME,PHONE,EMAILID
    FROM myuser;
</select>

<select id="getUserById" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
    FROM myuser
    WHERE USER_ID=#{user_id}
</select>

<update id="updateUser" parameterType="user">
    UPDATE myuser
    SET
    USERNAME = #{username},
    PASSWORD = #{password},
    FIRSTNAME= #{firstName},
    LASTNAME = #{lastName},
    PHONE    = #{phone},
    EMAILID  = #{emailId}
    WHERE
    USER_ID = #{user_id}
</update>

<delete id="deleteUser" parameterType="String">
    DELETE FROM
    myuser
    WHERE USER_ID = #{user_id}
</delete>

<select id="getUserByUsername" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
    FROM myuser
    WHERE USERNAME = #{username}
</select>
</mapper>
```

User.java:

```
package com.tutorials4u.model;
import org.apache.struts.action.ActionForm;

public class User extends ActionForm {

    private static final long serialVersionUID = 1L;
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }
}
```

```
public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.log4j.Logger;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.actions.MappingDispatchAction;
import com.tutorials4u.business.BusinessDelegate;
import com.tutorials4u.model.User;

public class UserAction extends MappingDispatchAction {
    // Get a logger
    private static final Logger logger = Logger
        .getLogger(UserAction.class);
    BusinessDelegate businessDelegate;

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }

    public void setBusinessDelegate(BusinessDelegate businessDeletage) {
        this.businessDelegate = businessDeletage;
    }

    public ActionForward listUsers(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response)
        throws Exception {
        logger.debug("UserAction.listUsers()");
        List<User> users=this.businessDelegate.listUsers();
        request.setAttribute("users", users);
        return mapping.findForward("success");
    }
}
```

```
}

public ActionForward insertUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    User user = (User) form;
    Logger.debug("UserAction.insertUser()");
    this.businessDelegate.insertUser(user);
    Logger.debug("1 Record Inserted Successfully " );
    return mapping.findForward("success");
}

public ActionForward deleteUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.deleteUser()");
    String userid = request.getQueryString();
    ArrayList<String> list = new ArrayList<String>();
    for (String newstr : userid.split("=")) {
        list.add(newstr);
    }
    Logger.debug("value=" + list.get(1));
    this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
    return mapping.findForward("success");
}

public ActionForward editUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    Logger.debug("UserAction.editUser()");
    String userid = request.getQueryString();
    ArrayList<String> list = new ArrayList<String>();
    for (String newstr : userid.split("=")) {
        list.add(newstr);
    }
    User user =
this.businessDelegate.editUser(Integer.parseInt(list.get(1)));
    request.setAttribute("user", user);
    return mapping.findForward("success");
}

public ActionForward updateUser(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response)
    throws Exception {
    User user = (User) form;
    Logger.debug("UserAction.updateUser()");
    this.businessDelegate.updateUser(user);
    Logger.debug("1 Record Updated Successfully");
    return mapping.findForward("success");
}

}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.service.UserService;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserService userDao = new UserService();

    public void insertUser(User user){
        logger.debug("BusinessDelegate.insertUser()");
        new UserService().insertUser(user);
    }

    public List<User> listUsers() {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.getAllUsers();
        return users;
    }

    public void deleteUser(int userid) {
        logger.debug("BusinessDelegate.deleteUser()");
        userDao.deleteUser(userid);
    }

    public User editUser(int userid){
        logger.debug("UserAction.editUser()");
        User user = userDao.getUserById(userid);
        return user;
    }

    public void updateUser(User user) {
        logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
    }
}
```

UserMapper.java:

```
package com.tutorials4u.mybatis.mappers;
import java.util.List;
import com.tutorials4u.model.User;

public interface UserMapper {
    public void insertUser(User user);
    public User getUserById(Integer userId);
    public List<User> getAllUsers();
    public void updateUser(User user);
    public void deleteUser(Integer userId);
}
```

UserService.java:

```
package com.tutorials4u.service;

import java.util.List;
import org.apache.ibatis.session.SqlSession;
import com.tutorials4u.model.User;
import com.tutorials4u.mybatis.MyBatisUtil;
import com.tutorials4u.mybatis.mappers.UserMapper;

public class UserService {

    public void insertUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.insertUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }

    public User getUserById(Integer userId) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            return userMapper.getUserById(userId);
        } finally {
            sqlSession.close();
        }
    }

    public List<User> getAllUsers() {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            return userMapper.getAllUsers();
        } finally {
            sqlSession.close();
        }
    }

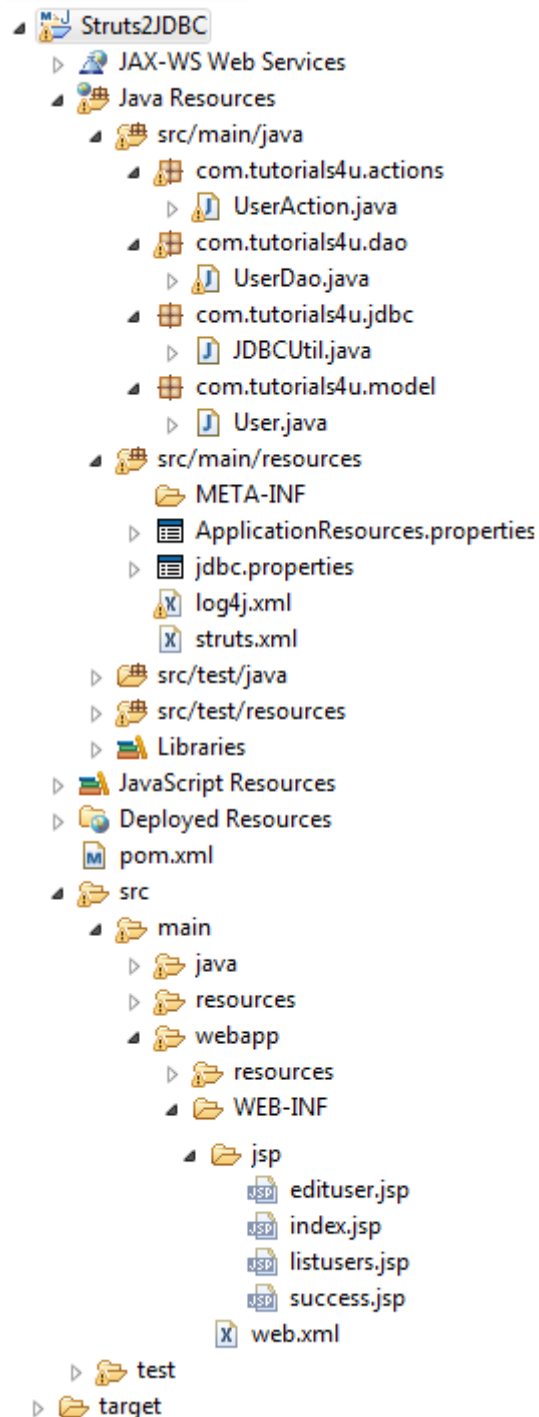
    public void updateUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.updateUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }
}
```

```
        }  
    }  
  
    public void deleteUser(Integer userId) {  
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()  
            .openSession();  
        try {  
            UserMapper userMapper =  
                sqlSession.getMapper(UserMapper.class);  
            userMapper.deleteUser(userId);  
            sqlSession.commit();  
        } finally {  
            sqlSession.close();  
        }  
    }  
}  
}
```

Index.jsp, edituser.jsp, listUsers.jsp are same as above Struts1 JDBC Integration

➤ Struts2 JDBC Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar</systemPath>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
    <display-name>Struts2 JDBC Integration</display-name>
    <filter>
        <filter-name>struts2jdbc</filter-name>
        <filter-class>
            org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
        </filter-class>
    </filter>
    <filter-mapping>
        <filter-name>struts2jdbc</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
</web-app>
```

Struts.xml :

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
    <include file="struts-default.xml"/>

    <constant name="struts.enable.DynamicMethodInvocation" value="true" />
    <constant name="struts.devMode" value="false" />
    <constant name="struts.custom.i18n.resources"
        value="ApplicationResources" />

    <package name="default" extends="struts-default">
        <action name="">
            <result>/WEB-INF/jsp/index.jsp</result>
        </action>

        <action name="insertAction" method="insertUser"
class="com.tutorials4u.actions.UserAction">
            <result name="input">/WEB-INF/jsp/index.jsp</result>
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="listUsersAction"
class="com.tutorials4u.actions.UserAction" method="listUsers">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="deleteUserAction"
class="com.tutorials4u.actions.UserAction" method="deleteUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="editUserAction"
class="com.tutorials4u.actions.UserAction" method="editUser">
            <result name="success">/WEB-INF/jsp/edituser.jsp</result>
        </action>

        <action name="updateUserAction"
class="com.tutorials4u.actions.UserAction" method="updateUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

    </package>
</struts>
```

jdbc.properties :

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

ApplicationResources.properties:

```
label.firstName = First Name
label.lastName = Last Name
label.username = Username
label.password = Password
label.phone = Phone
label.emailId = Email Id
```

log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
  xmlns:log4j='http://jakarta.apache.org/Log4j/'>

  <!-- Console -->
  <appender name="console" class="org.apache.Log4j.ConsoleAppender">
    <layout class="org.apache.Log4j.PatternLayout">
      <param name="ConversionPattern"
        value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
  </appender>

  <!-- file -->
  <appender name="file" class="org.apache.log4j.RollingFileAppender">
    <param name="append" value="false" />
    <param name="maxFileSize" value="10KB" />
    <param name="maxBackupIndex" value="5" />
    <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
  />
    <layout class="org.apache.Log4j.PatternLayout">
      <param name="ConversionPattern"
        value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
  </appender>

  <root>
    <level value="DEBUG" />
    <appender-ref ref="console" />
    <appender-ref ref="file" />
  </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;

public class User{

    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
```

```
        this.phone = phone;
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }

    @Override
    public String toString() {
        return "Id: " + user_id + "; fname : " + firstName + "' phone : "
            + phone + "; emailId: " + emailId;
    }
}
```

UserAction.java:

```
package com.tutorials4u.actions;
import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserDao userDao = new UserDao();
    private User user;
    HttpServletRequest request;

    private Object users, editusers;

    public Object getUsers() {
        return users;
    }

    public void setUsers(Object users) {
        this.users = users;
    }

    public Object getEditusers() {
        return editusers;
    }

    public void setEditusers(Object editusers) {
        this.editusers = editusers;
    }

    public User getUser(){
        return user;
    }
}
```

```
public void setUser(User user){
    this.user = user;
}

public void setServletRequest(HttpServletRequest request) {
    this.request = request;
}

public HttpServletRequest getServletRequest() {
    return this.request;
}

public String insertUser() throws Exception {
    Logger.debug("UserAction.insertUser()");
    String result = new UserDao().insertUser(user);
    Logger.debug("1 Record Inserted Successfully " + result);
    return ActionSupport.SUCCESS;
}

public String listUsers()
    throws Exception {
    Logger.debug("UserAction.listUsers()");
    users = userDao.listUsers();
    return ActionSupport.SUCCESS;
}

public String deleteUser()
    throws Exception {
    Logger.debug("UserAction.deleteUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    Logger.debug("value="+list.get(1));
    userDao.deleteUser(Integer.parseInt(list.get(1)));
    Logger.debug("1 Record Deleted Successfully ");
    return ActionSupport.SUCCESS;
}

public String editUser()
    throws Exception {
    Logger.debug("UserAction.editUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    editusers=userDao.editUser(Integer.parseInt(list.get(1)));
    return ActionSupport.SUCCESS;
}

public String updateUser()
    throws Exception {
    Logger.debug("UserAction.updateUser()");
    userDao.updateUser(user);
}
```

```
        logger.debug("1 Record Updated Successfully");
        return ActionSupport.SUCCESS;
    }
}
```

JDBCUtil.java:

```
package com.tutorials4u.jdbc;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.util.Properties;

public class JDBCUtil {
    private static Connection connection = null;
    InputStream inputStream;
    String propFileName = "jdbc.properties";
    public static Properties prop = new Properties();
    public JDBCUtil() throws IOException {
        inputStream = getClass().getClassLoader().getResourceAsStream(
            propFileName);
        if (inputStream != null) {
            prop.load(inputStream);
        } else {
            throw new FileNotFoundException("property file '" +
propFileName
                + "' not found in the classpath");
        }
    }

    public static Connection getMysqlConnection() throws IOException {
        String driverClass = prop.getProperty("jdbc.driverClassName");
        String url = prop.getProperty("jdbc.url");
        String username = prop.getProperty("jdbc.username");
        String password = prop.getProperty("jdbc.password");

        if (connection == null) {
            try {
                Class.forName(driverClass).newInstance();
                connection = DriverManager.getConnection(url, username,
                    password);
                System.out.println("Connected To Mysql Datasbase");
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        return connection;
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;

import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.jdbc.JDBCUtil;
import com.tutorials4u.model.User;

public class UserDao {
    private static final Logger logger = Logger.getLogger(UserAction.class);

    PreparedStatement stmt = null;

    @SuppressWarnings("static-access")
    public String insertUser(User newuser) throws IOException, SQLException {
        logger.debug("UserDao.createUser()");
        JDBCUtil util = new JDBCUtil();
        Connection conn = util.getMysqlConnection();
        try {
            String username = newuser.getUsername();
            String password = newuser.getPassword();
            String firstname = newuser.getFirstName();
            String lastname = newuser.getLastName();
            String emailId = newuser.getEmailId();
            String phone = newuser.getPhone();

            String sql = new String(
                "insert into
myuser(USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID) values(?,?,?,?,?,?)");

            stmt = conn.prepareStatement(sql);
            stmt.setString(1, username);
            stmt.setString(2, password);
            stmt.setString(3, firstname);
            stmt.setString(4, lastname);
            stmt.setString(5, phone);
            stmt.setString(6, emailId);

            logger.debug("Inserting values...");
            stmt.executeUpdate();
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            return "failure";
        } finally {
            stmt.close();
            conn.close();
        }
        return "success";
    }
}
```



```
@SuppressWarnings("static-access")
public List<User> listUsers() throws IOException, SQLException {
    Logger.debug("UserDao.listUsers()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    List<User> users = new ArrayList<User>();
    try {
        String sql = "select
USER_ID,USERNAME,FIRSTNAME,LASTNAME,PHONE,EMAILID from myuser";
        statement = conn.createStatement();
        result = statement.executeQuery(sql);
        User newuser = null;
        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
            users.add(newuser);
        }
    } catch (SQLException e) {
        Logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
        // conn.close();
    }
    return users;
}

@SuppressWarnings("static-access")
public void deleteUser(int id) throws IOException, SQLException {
    Logger.debug("UserDao.deleteUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    Statement statement = null;
    try {
        String sql = "delete from myuser where user_id=" + id;
        statement = conn.createStatement();
        statement.executeUpdate(sql);
    } catch (SQLException e) {
        Logger.error("Error deleting User :" + e.getMessage());
    } finally {
        statement.close();
    }
}

@SuppressWarnings("static-access")
public User editUser(int id) throws IOException, SQLException {
    Logger.debug("UserDao.editUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    User user = getUser(id);
    return user;
}
```

```
@SuppressWarnings("static-access")
public User getUser(int id) throws SQLException, IOException {
    Logger.debug("UserDao.getUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    User newuser = null;
    try {
        String sql = "select
USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID from myuser WHERE
USER_ID="
                    + id;
        statement = conn.createStatement();
        result = statement.executeQuery(sql);

        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setPassword(result.getString("PASSWORD"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
        }
    } catch (SQLException e) {
        Logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
        // conn.close();
    }
    return newuser;
}

@SuppressWarnings("static-access")
public void updateUser(User newuser) throws IOException {
    Logger.debug("UserDao.updateUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    try {
        int userid = newuser.getUser_id();
        String username = newuser.getUsername();
        String password = newuser.getPassword();
        String firstname = newuser.getFirstName();
        String lastname = newuser.getLastName();
        String emailId = newuser.getEmailId();
        String phone = newuser.getPhone();

        String sql = "UPDATE MYUSER SET
USERNAME=?,PASSWORD=?,FIRSTNAME=?,LASTNAME=?,PHONE=?,EMAILID=? WHERE
USER_ID="+userid;

        stmt = conn.prepareStatement(sql);
        stmt.setString(1, username);
        stmt.setString(2, password);
        stmt.setString(3, firstname);
```

```
        stmt.setString(4, lastname);
        stmt.setString(5, phone);
        stmt.setString(6, emailId);

        logger.debug("Updating values....");
        stmt.executeUpdate();
    } catch (Exception e) {
        logger.error("Error updating User :" + e.getMessage());
    }
}

}
```

Index.jsp:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="s" uri="/struts-tags" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert User Details</title>
<script type="text/javascript"
    src="<c:url value='/resources/mytheme/js/customalert.js' />"></script>
<link rel="stylesheet" type="text/css"
    href="<c:url value='/resources/mytheme/css/customalert.css' />" />
<script type="text/javascript">

    function setColor(el, bg)
    {
        if (el.style) el.style.backgroundColor = bg;
    }

    function checkUserName(form)
    {
        var bgBad = "yellow";
        var bgGood = "lightyellow";
        var valid = true;
        var errmsg="";

        if (form.username.value == "" ||
form.username.value.trim().length==0) {
            valid = false;
            setColor(form.username, bgBad);

        } else {
            setColor(form.username, bgGood);
        }

        errmsg=errmsg+"\n\nPlease enter the Username ";
        if (!valid)
            createCustomAlert(errmsg,"","Errors");
        return valid;
    }

}
```

```
function checkInput(form)
{
    var bgBad = "yellow";
    var bgGood = "lightyellow";
    var valid = true;
    var errmsg="";

    if (form.firstName.value == "" ||
form.firstName.value.trim().length==0) {
        valid = false;
        setColor(form.firstName, bgBad);
    } else {
        setColor(form.firstName, bgGood);
    }

    if (form.lastName.value == "" ||
form.lastName.value.trim().length==0) {
        valid = false;
        setColor(form.lastName, bgBad);
    } else {
        setColor(form.lastName, bgGood);
    }

    if (form.username.value == "" ||
form.username.value.trim().length==0) {
        valid = false;
        setColor(form.username, bgBad);
    } else {
        setColor(form.username, bgGood);
    }

    if (form.password.value == "" ||
form.password.value.trim().length==0) {
        valid = false;
        setColor(form.password, bgBad);
    } else {
        setColor(form.password, bgGood);
    }

    if (form.phone.value == "" ||
form.phone.value.trim().length==0) {
        valid = false;
        setColor(form.phone, bgBad);
    } else {
        setColor(form.phone, bgGood);
    }

    if (form.phone.value.trim().length < 10)
    {
        valid = false;
        setColor(form.phone, bgBad);
    }
}
```

```
        createCustomAlert("Please Enter a valid Contact
Number", "", "Errors");
        form.phone.value="";
    }
    else
    {
        setColor(form.phone, bgGood);
    }

    if(document.addForm.emailId.value=="")
    {
        createCustomAlert("Please Enter a valid E-mail
Address", "", "Errors");
        document.addForm.emailId.focus();
        return false;
    }
    //E-mail field should accept valid characters only.
    if (document.addForm.emailId.value != " ")
    {
        var
checkOK="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-@._";
        var checkStr = document.addForm.emailId.value;
        var allValid = 'true';
        for (i = 0; i < checkStr.length+1; i++)
        {
            ch = checkStr.charAt(i);
            for (j = 0; j < checkOK.length+1; j++)
            {
                if (ch == checkOK.charAt(j))
                    break;
                if (j == checkOK.length)
                {
                    allValid = 'false';
                    break;
                }
            }
        }
    }
    //E-mail id must have '@'
    if(document.addForm.emailId.value != "")
    {
        var val=document.addForm.emailId.value ;
        var pass1=val.indexOf('@',0);
        if(pass1!=-1)
        {
            //alert("Not a valid E-Mail Address");
            createCustomAlert("Not a valid E-Mail
Address", "", "Errors");

            document.addForm.emailId.value="";
            document.addForm.emailId.focus();
            return false;
        }
    }
    //E-mail id must have '.'
    if(document.addForm.emailId.value != "")
    {
        var val=document.addForm.emailId.value ;
```

```

        var pass2=val.indexOf('.',0);
        if(pass2==-1)
        {
            //alert("Not a valid E-Mail Address");
            createCustomAlert("Not a valid E-Mail
Address", "", "Errors");

            document.addForm.emailId.value="";
            document.addForm.emailId.focus();
            return false;
        }
        if((pass2 - pass1)== 1)
        {
            //alert("Not a valid E-Mail Address");
            createCustomAlert("Not a valid E-Mail
Address", "", "Errors");

            document.addForm.emailId.value="";
            document.addForm.emailId.focus();
            return false;
        }
    }
    //E-mail id should have only @,.,_ and - with the letters and
digits
    if (!allValid)
    {
        createCustomAlert("Please enter only letters,
digits,and \"@\" \"_\" \".\" \"-\" characters in the \"EMail\"
field.", "Errors");

        document.addForm.emailId.focus();
        return false;
    }

    errmsg=errmsg+"\n\nPlease enter the fields marked with * ";

    if (!valid)
        createCustomAlert(errmsg, "", "Errors");

    return valid;
}

function isNumberKey(evt)
{
    var chCode = (evt.which) ? evt.which : event.keyCode
    if (chCode != 46 && chCode > 31
        && (chCode < 48 || chCode > 57))
        return false;
    else
        return true;
}

function isAlpha(keyCode)
{
    return ((keyCode >= 65 && keyCode <= 90) || keyCode == 8
|| keyCode == 32 || keyCode == 9 || keyCode == 46)
}

</script>

</head>

```

```
<body id="dt_example">
  <b>Struts2 + JDBC Integration</b>
  <div class="full_w">
    <s:if test="hasActionErrors()">
      <div id="fieldErrors">
        <s:actionerror/>
      </div>
    </s:if>

    <s:form action="insertAction" namespace="/" method="post"
name="strutsForm">
      <div class="element">
        <s:hidden name="user_id" />
      </div>

      <div class="element">
        <s:textfield name="user.firstName" size="30"
maxlength="50" key="Label.firstName"/>
      </div>

      <div class="element">
        <s:textfield name="user.lastName" size="30"
maxlength="50" key="Label.lastName"/>
      </div>

      <div class="element">
        <s:textfield name="user.username" size="30" maxlength="50"
key="Label.username"/>
      </div>

      <div class="element">
        <s:password name="user.password" size="30"
maxlength="50" key="Label.password"/>
      </div>

      <div class="element">
        <s:textfield name="user.phone" size="30"
maxlength="50" key="Label.phone"/>
      </div>

      <div class="element">
        <s:textfield name="user.emailId" size="30"
maxlength="50" key="Label.emailId"/>
      </div>

      <div class="entry">
        <s:submit key="Submit" align="right"/>
      </div>
    </s:form>
  </div>

</body>
</html>
```

ListOfUsers.jsp:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
```

```

        pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="s" uri="/struts-tags" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>List Of Users</title>
<link href="<c:url value='/resources/mytheme/css/demo_table.css' />"
rel="stylesheet" type="text/css" />
<link href="<c:url value='/resources/mytheme/css/demo_table_jui.css' />"
rel="stylesheet" type="text/css" />
<link href="<c:url value='/resources/mytheme/css/jquery-ui-1.8.24.custom.css' />"
rel="stylesheet" type="text/css" media="all" />
<script src="<c:url value='/resources/mytheme/js/jquery.js' />"
type="text/javascript"></script>
<script src="<c:url value='/resources/mytheme/js/jquery.dataTables.js' />"
type="text/javascript"></script>
<script type="text/javascript">
    $(document).ready(function() {
        $("#companies").dataTable({
            "sPaginationType" : "full_numbers",
            "bJQueryUI" : true
        });
    });
</script>
</head>
<body id="dt_example">
    <div class="full_w">
        <div class="h_title">
            &nbsp;&nbsp;&nbsp;List Of Users
        </div>
        <br>
        <br>
        <br>
        <table id="companies" class="display">
            <thead>
                <tr>
                    <th>Username</th>
                    <th><u>First Name</u></th>
                    <th><u>Last Name</u></th>
                    <th><u>Phone</u></th>
                    <th><u>Email Id</u></th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                <s:iterator value="users">
                    <tr>
                        <td align='center'> <s:property value =
"username"/></td>
                        <td align='center'> <s:property value =
"firstName"/></td>
                        <td align='center'> <s:property value =
"lastName"/></td>
                        <td align='center'> <s:property value =
"phone"/></td>

```



```

                                <td align='center'> <s:property value =
"emailId"/></td>
                                <td align='center'><a
                                href="/Struts2JDBC/editUserAction.action?userid=<s:property value =
'userid'>"></a>&nbsp;&nbsp;&nbsp;<a
                                href="/Struts2JDBC/deleteUserAction.action?userid=<s:property value =
'userid'>"></a></td>
                                </tr>
                                </s:iterator>
                                </tbody>
                                </table>
                                </div>
</body>
</html>
```

editUser.jsp:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="s" uri="/struts-tags"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Edit User Details</title>
<script type="text/javascript" src="<c:url
value='/resources/mytheme/js/customalert.js'>"></script>
<link rel="stylesheet" type="text/css" href="<c:url
value='/resources/mytheme/css/customalert.css'>" />

<script type="text/javascript">

    function setColor(el, bg)
    {
        if (el.style) el.style.backgroundColor = bg;
    }

    function checkUserName(form)
    {
        var bgBad = "yellow";
        var bgGood = "lightyellow";
        var valid = true;
        var errmsg="";

        if (form.username.value == "" ||
form.username.value.trim().length==0) {
            valid = false;
            setColor(form.username, bgBad);
        } else {
            setColor(form.username, bgGood);
        }
    }
</script>
```

```
    }

    errmsg=errmsg+"\n\nPlease enter the Username ";

    if (!valid)
        createCustomAlert(errmsg, "", "Errors");

    return valid;
}

function checkInput(form)
{
    var bgBad = "yellow";
    var bgGood = "lightyellow";
    var valid = true;
    var errmsg="";

    if (form.firstName.value == "" ||
form.firstName.value.trim().length==0) {
        valid = false;
        setColor(form.firstName, bgBad);
    } else {
        setColor(form.firstName, bgGood);
    }

    if (form.lastName.value == "" ||
form.lastName.value.trim().length==0) {
        valid = false;
        setColor(form.lastName, bgBad);
    } else {
        setColor(form.lastName, bgGood);
    }

    if (form.username.value == "" ||
form.username.value.trim().length==0) {
        valid = false;
        setColor(form.username, bgBad);
    } else {
        setColor(form.username, bgGood);
    }

    if (form.password.value == "" ||
form.password.value.trim().length==0) {
        valid = false;
        setColor(form.password, bgBad);
    } else {
        setColor(form.password, bgGood);
    }

    if (form.phone.value == "" || form.phone.value.trim().length==0)
{
        valid = false;
    }
}
```

```
        setColor(form.phone, bgBad);

    } else {
        setColor(form.phone, bgGood);
    }

    if (form.phone.value.trim().length < 10)
    {
        valid = false;
        setColor(form.phone, bgBad);
        createCustomAlert("Please Enter a valid Contact
Number", "", "Errors");
        form.phone.value="";
    }
    else
    {
        setColor(form.phone, bgGood);
    }

    if(document.addForm.emailId.value=="")
    {
        createCustomAlert("Please Enter a valid E-mail
Address", "", "Errors");
        document.addForm.emailId.focus();
        return false;
    }
    //E-mail field should accept valid characters only.
    if (document.addForm.emailId.value != " ")
    {
        var
checkOK="ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-@._";
        var checkStr = document.addForm.emailId.value;
        var allValid = 'true';
        for (i = 0; i < checkStr.length+1; i++)
        {
            ch = checkStr.charAt(i);
            for (j = 0; j < checkOK.length+1; j++)
            {
                if (ch == checkOK.charAt(j))
                    break;
                if (j == checkOK.length)
                {
                    allValid = 'false';
                    break;
                }
            }
        }
    }
    //E-mail id must have '@'
    if(document.addForm.emailId.value != "")
    {
        var val=document.addForm.emailId.value ;
        var pass1=val.indexOf('@',0);
        if(pass1!=-1)
        {
            //alert("Not a valid E-Mail Address");
            createCustomAlert("Not a valid E-Mail
Address", "", "Errors");
        }
    }
}
```

```

        document.addForm.emailId.value="";
        document.addForm.emailId.focus();
        return false;
    }
}
//E-mail id must have '.'
if(document.addForm.emailId.value != "")
{
    var val=document.addForm.emailId.value ;
    var pass2=val.indexOf('.',0);
    if(pass2== -1)
    {
        //alert("Not a valid E-Mail Address");
        createCustomAlert("Not a valid E-Mail
Address", "", "Errors");

        document.addForm.emailId.value="";
        document.addForm.emailId.focus();
        return false;
    }
    if((pass2 - pass1)== 1)
    {
        //alert("Not a valid E-Mail Address");
        createCustomAlert("Not a valid E-Mail
Address", "", "Errors");

        document.addForm.emailId.value="";
        document.addForm.emailId.focus();
        return false;
    }
}
//E-mail id should have only @,.,_ and - with the letters and
digits
if (!allValid)
{
    createCustomAlert("Please enter only letters,
digits,and \"@\" \"_\" \".\" \"-\" characters in the \"EMail\"
field.", "Errors");
    document.addForm.emailId.focus();
    return false;
}

errmsg=errmsg+"\n\nPlease enter the fields marked with * ";

if (!valid)
    createCustomAlert(errmsg, "", "Errors");

return valid;
}

function isNumberKey(evt)
{
    var chCode = (evt.which) ? evt.which : event.keyCode
    if (chCode != 46 && chCode > 31
        && (chCode < 48 || chCode > 57))
        return false;
    else
        return true;
}

```

```
function isAlpha(keyCode)
{
    return ((keyCode >= 65 && keyCode <= 90) || keyCode == 8
|| keyCode == 32 || keyCode == 9 || keyCode == 46)
}
</script>
</head>
<body id="dt_example">
    <b>Struts2 + JDBC Integration</b>
    <br>
    <br>
    <div class="full_w">
        <s:form action="updateUserAction" namespace="/" method="post"
            name="strutsForm">
            <div class="element">
                <s:hidden name="user.user_id"
value="%{editusers.user_id}" />
            </div>

            <div class="element">
                <s:textfield name="user.firstName" size="30"
maxlength="50"
                    key="label.firstName"
value="%{editusers.firstName}" />
            </div>

            <div class="element">
                <s:textfield name="user.lastName" size="30"
maxlength="50"
                    key="label.lastName"
value="%{editusers.lastName}" />
            </div>

            <div class="element">
                <s:textfield name="user.username" size="30"
maxlength="50"
                    key="label.username"
value="%{editusers.username}" />
            </div>

            <div class="element">
                <s:password name="user.password" size="30"
maxlength="50"
                    key="label.password"
value="%{editusers.password}" />
            </div>

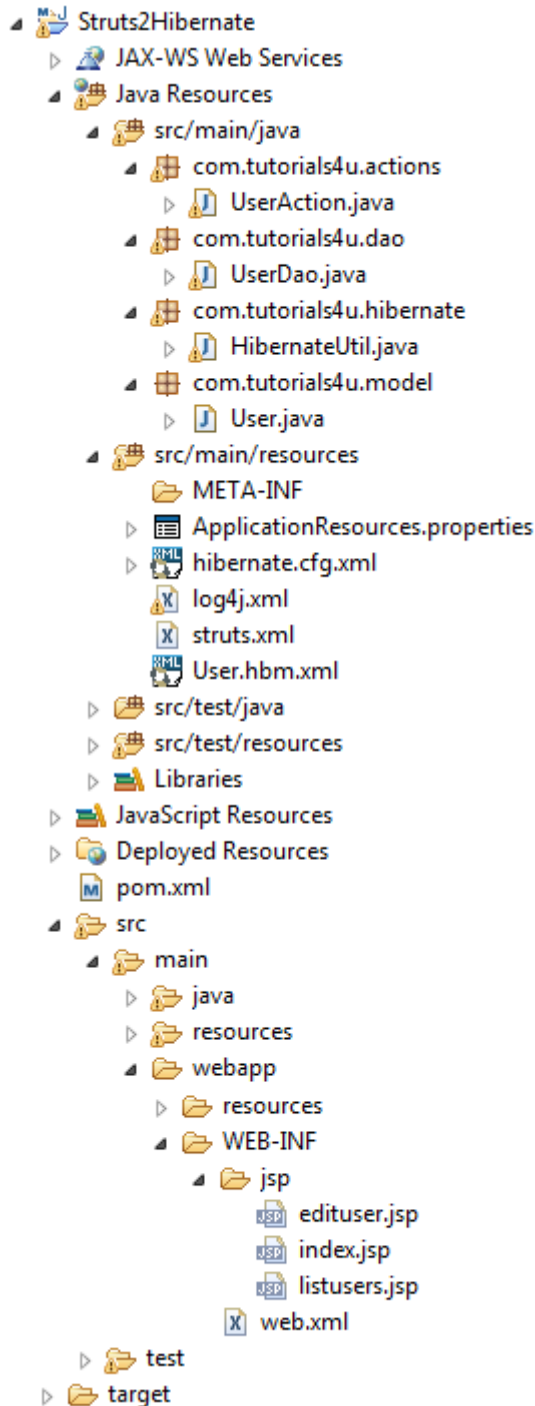
            <div class="element">
                <s:textfield name="user.phone" size="30" maxlength="50"
                    key="label.phone" value="%{editusers.phone}" />
            </div>

            <div class="element">
                <s:textfield name="user.emailId" size="30"
maxlength="50"
                    key="label.emailId" value="%{editusers.emailId}"
/>
            </div>
        </s:form>
    </div>
```

```
        <div class="entry">
            <s:submit key="Submit" align="right" />
        </div>
    </s:form>
</div>
</body>
</html>
```

➤ Struts2 Hibernate Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Hibernate framework -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>4.2.4.Final</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar
    </systemPath>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
    <display-name>Struts2 Hibernate Integration</display-name>
    <filter>
        <filter-name>struts2hibernate</filter-name>
        <filter-class>
            org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
        </filter-class>
    </filter>
    <filter-mapping>
        <filter-name>struts2hibernate</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
</web-app>
```


Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <include file="struts-default.xml"/>

  <constant name="struts.enable.DynamicMethodInvocation" value="true" />
  <constant name="struts.devMode" value="false" />
  <constant name="struts.custom.i18n.resources" value="ApplicationResources"
/>

  <package name="default" extends="struts-default">
    <action name="">
      <result>/WEB-INF/jsp/index.jsp</result>
    </action>

    <action name="insertAction" method="insertUser"
class="com.tutorials4u.actions.UserAction">
      <result name="input">/WEB-INF/jsp/index.jsp</result>
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="listUsersAction"
class="com.tutorials4u.actions.UserAction" method="listUsers">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="deleteUserAction"
class="com.tutorials4u.actions.UserAction" method="deleteUser">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="editUserAction"
class="com.tutorials4u.actions.UserAction" method="editUser">
      <result name="success">/WEB-INF/jsp/edituser.jsp</result>
    </action>

    <action name="updateUserAction"
class="com.tutorials4u.actions.UserAction" method="updateUser">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

  </package>
</struts>
```

ApplicationResources.properties :

```
label.firstName = First Name
label.lastName  = Last Name
label.username  = Username
label.password  = Password
label.phone     = Phone
label.emailId   = Email Id
```

hibernate.cfg.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property name="hibernate.connection.driver_class">
            com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.url">
            jdbc:mysql://localhost:3306/hibernate</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password"></property>
        <property name="hibernate.dialect">
            org.hibernate.dialect.MySQLDialect</property>
        <property name="show_sql">true</property>
        <property name="format_sql">true</property>
        <property name="hbm2ddl.auto">update</property>
        <mapping resource="User.hbm.xml" />
    </session-factory>
</hibernate-configuration>
```

User.hbm.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
"-//Hibernate/Hibernate Mapping DTD//EN"
"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.tutorials4u.model.User" table="MYUSER">
        <meta attribute="class-description">
            This class contains the User detail.
        </meta>
        <id name="user_id" type="int" column="USER_ID">
            <generator class="native" />
        </id>
        <property name="username" column="USERNAME" type="string" />
        <property name="password" column="PASSWORD" type="string" />
        <property name="firstName" column="FIRSTNAME" type="string" />
        <property name="lastName" column="LASTNAME" type="string" />
        <property name="phone" column="PHONE" type="string" />
        <property name="emailId" column="EMAILID" type="string" />
    </class>
</hibernate-mapping>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
    />
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;

import javax.persistence.Transient;
public class User{

    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
        this.valid = valid;
    }
}
```

```
public int getUser_id() {
    return user_id;
}

public void setUser_id(int user_id) {
    this.user_id = user_id;
}

public String getUsername() {
    return username;
}

public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}
```

```
@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserDao userDao = new UserDao();
    private User user;
    HttpServletRequest request;

    private Object users, editusers;

    public Object getUsers() {
        return users;
    }

    public void setUsers(Object users) {
        this.users = users;
    }

    public Object getEditusers() {
        return editusers;
    }

    public void setEditusers(Object editusers) {
        this.editusers = editusers;
    }

    public User getUser(){
        return user;
    }

    public void setUser(User user){
        this.user = user;
    }

    public void setServletRequest(HttpServletRequest request) {
        this.request = request;
    }
}
```

```
public HttpServletRequest getServletRequest() {
    return this.request;
}

public String insertUser() throws Exception {
    logger.debug("UserAction.insertUser()");
    String result = new UserDao().insertUser(user);
    logger.debug("1 Record Inserted Successfully " + result);
    return ActionSupport.SUCCESS;
}

public String listUsers()
    throws Exception {
    logger.debug("UserAction.listUsers()");
    users = userDao.listUsers();
    return ActionSupport.SUCCESS;
}

public String deleteUser()
    throws Exception {
    logger.debug("UserAction.deleteUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    logger.debug("value="+list.get(1));
    userDao.deleteUser(Integer.parseInt(list.get(1)));
    logger.debug("1 Record Deleted Successfully ");
    return ActionSupport.SUCCESS;
}

public String editUser()
    throws Exception {
    logger.debug("UserAction.editUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    editusers=userDao.editUser(Integer.parseInt(list.get(1)));
    return ActionSupport.SUCCESS;
}

public String updateUser()
    throws Exception {
    logger.debug("UserAction.updateUser()");
    userDao.updateUser(user);
    logger.debug("1 Record Updated Successfully");
    return ActionSupport.SUCCESS;
}
}
```

HibernateUtil.java:

```
package com.tutorials4u.hibernate;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class HibernateUtil {

    private static final SessionFactory sessionFactory =
buildSessionFactory();

    private static SessionFactory buildSessionFactory() {
        try {
            // Create the SessionFactory from hibernate.cfg.xml
            return new Configuration().configure().buildSessionFactory();
        }
        catch (Throwable ex) {
            // Make sure you log the exception, as it might be swallowed
            System.err.println("Initial SessionFactory creation failed." +
ex);

            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

    public static void shutdown() {
        // Close caches and connection pools
        getSessionFactory().close();
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;

import java.util.List;

import org.hibernate.Session;
import org.apache.log4j.Logger;

import com.tutorials4u.actions.UserAction;
import com.tutorials4u.hibernate.HibernateUtil;
import com.tutorials4u.model.User;

public class UserDao {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);

    public String insertUser(User newuser) {
        logger.debug("UserDao.createUser()");
        Session session = HibernateUtil.getSessionFactory().openSession();
        try {
            session.beginTransaction();
            User user = new User();
            user.setUsername(newuser.getUsername());
        }
```

```
        user.setPassword(newuser.getPassword());
        user.setFirstName(newuser.getFirstName());
        user.setLastName(newuser.getLastName());
        user.setEmailId(newuser.getEmailId());
        user.setPhone(newuser.getPhone());
        logger.debug("Inserting values...");
        session.save(user);
        session.getTransaction().commit();
    } catch (Exception e) {
        logger.error("Error creating User :" + e.getMessage());
        session.getTransaction().rollback();
        return "failure";
    }
    return "success";
}

@SuppressWarnings("unchecked")
public List<User> listUsers() {
    logger.debug("UserDao.listUsers()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    return session.createQuery("From User").list();
}

public void deleteUser(int id) {
    logger.debug("UserDao.deleteUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    User user = getUser(id);
    session.delete(user);
    session.getTransaction().commit();
}

public User editUser(int id) {
    logger.debug("UserDao.editUser()");
    @SuppressWarnings("unused")
    Session session = HibernateUtil.getSessionFactory().openSession();
    User user = getUser(id);
    return user;
}

public User getUser(int id) {
    logger.debug("UserDao.getUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    User user = (User) session.get(User.class, id);
    return user;
}

public void updateUser(User user) {
    logger.debug("UserDao.updateUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.update(user);
    session.getTransaction().commit();
}
}
```

Index.jsp,edituser.jsp and listUsers.jsp are same as Struts2 JDBC Integration project.

➤ Struts2 iBatis Integration

Project Structure:

- └─ Struts2Ibatis
 - └─ JAX-WS Web Services
 - └─ Deployment Descriptor: Struts2Ibatis
 - └─ Java Resources
 - └─ src/main/java
 - └─ com.tutorials4u.actions
 - └─ UserAction.java
 - └─ com.tutorials4u.dao
 - └─ UserDao.java
 - └─ com.tutorials4u.ibatis
 - └─ IbatisUtil.java
 - └─ com.tutorials4u.model
 - └─ User.java
 - └─ src/main/resources
 - └─ META-INF
 - └─ ApplicationResources.properties
 - └─ log4j.xml
 - └─ sql-maps-config.xml
 - └─ struts.xml
 - └─ user.xml
 - └─ src/test/java
 - └─ src/test/resources
 - └─ Libraries
 - └─ JavaScript Resources
 - └─ Deployed Resources
 - └─ pom.xml
 - └─ src
 - └─ main
 - └─ java
 - └─ resources
 - └─ webapp
 - └─ resources
 - └─ WEB-INF
 - └─ jsp
 - └─ edituser.jsp
 - └─ index.jsp
 - └─ listusers.jsp
 - └─ success.jsp
 - └─ web.xml
 - └─ test
 - └─ target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Ibatis -->
<dependency>
    <groupId>org.apache.ibatis</groupId>
    <artifactId>ibatis-sqlmap</artifactId>
    <version>2.3.4.726</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar
    </systemPath>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
    <display-name>Struts2 iBatis Integration</display-name>
    <filter>
        <filter-name>struts2ibatis</filter-name>
        <filter-class>
            org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
        </filter-class>
    </filter>
    <filter-mapping>
        <filter-name>struts2ibatis</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
</web-app>
```

Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <include file="struts-default.xml"/>
  <constant name="struts.enable.DynamicMethodInvocation" value="true" />
  <constant name="struts.devMode" value="false" />
  <constant name="struts.custom.i18n.resources" value="ApplicationResources"
/>

  <package name="default" extends="struts-default">
    <action name="">
      <result>/WEB-INF/jsp/index.jsp</result>
    </action>

    <action name="insertAction" method="insertUser"
class="com.tutorials4u.actions.UserAction">
      <result name="input">/WEB-INF/jsp/index.jsp</result>
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="listUsersAction"
class="com.tutorials4u.actions.UserAction" method="listUsers">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="deleteUserAction"
class="com.tutorials4u.actions.UserAction" method="deleteUser">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="editUserAction"
class="com.tutorials4u.actions.UserAction" method="editUser">
      <result name="success">/WEB-INF/jsp/edituser.jsp</result>
    </action>

    <action name="updateUserAction"
class="com.tutorials4u.actions.UserAction" method="updateUser">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

  </package>
</struts>
```

ApplicationResources.properties:

```
label.firstName = First Name
label.lastName = Last Name
label.username = Username
label.password = Password
label.phone = Phone
label.emailId = Email Id
```

sql-maps-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE sqlMapConfig
    PUBLIC "-//ibatis.apache.org//DTD SQL Map Config 2.0//EN"
    "http://ibatis.apache.org/dtd/sql-map-config-2.dtd">
<sqlMapConfig>
    <settings useStatementNamespaces="true"/>
    <transactionManager type="JDBC">
        <dataSource type="SIMPLE">
            <property name="JDBC.Driver" value="com.mysql.jdbc.Driver"/>
            <property name="JDBC.ConnectionURL"
value="jdbc:mysql://localhost:3306/hibernate"/>
            <property name="JDBC.Username" value="root"/>
            <property name="JDBC.Password" value=""/>
        </dataSource>
    </transactionManager>
    <sqlMap resource="user.xml"/>
</sqlMapConfig>
```

User.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sqlMap PUBLIC "-//ibatis.apache.org//DTD SQL Map 2.0//EN"
"http://ibatis.apache.org/dtd/sql-map-2.dtd">
<sqlMap namespace="user">

    <typeAlias alias="USER" type="com.tutorials4u.model.User" />

    <resultMap id="userResultMap" class="USER">
        <result property="user_id" column="USER_ID" />
        <result property="username" column="USERNAME" />
        <result property="password" column="PASSWORD" />
        <result property="firstName" column="FIRSTNAME" />
        <result property="lastName" column="LASTNAME" />
        <result property="phone" column="PHONE" />
        <result property="emailId" column="EMAILID" />
    </resultMap>

    <select id="listUsers" resultMap="userResultMap">
        SELECT * FROM MYUSER
    </select>

    <insert id="insertUser" parameterClass="USER">
        INSERT INTO MYUSER
        (USERNAME,PASSWORD,FIRSTNAME, LASTNAME,PHONE,EMAILID)
```

```
VALUES(#username#,#password#,#firstName#,#lastName#,#phone#,#emailId#);
</insert>

<delete id="deleteUser" parameterClass="java.Lang.Integer">
    DELETE FROM MYUSER WHERE USER_ID = #value#
</delete>

<select id="getUserById" parameterClass="java.Lang.Integer"
    resultMap="userResultMap">
    SELECT * FROM MYUSER WHERE USER_ID = #value#
</select>

<update id="updateUser" parameterClass="USER">
    UPDATE MYUSER
    SET USERNAME = #username#,
    PASSWORD     = #password#,
    FIRSTNAME    = #firstName#,
    LASTNAME     = #lastName#,
    PHONE        = #phone#,
    EMAILID      = #emailId#
    WHERE USER_ID = #user_id#
</update>

</sqlMap>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>
    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.log"
    />

        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;

public class User{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }
}
```

```
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }

    @Override
    public String toString() {
        return "Id: " + user_id + "; fname : " + firstName + "' phone : "
            + phone + "; emailId: " + emailId;
    }
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserDao userDao = new UserDao();
    private User user;
    HttpServletRequest request;

    private Object users, editusers;

    public Object getUsers() {
        return users;
    }

    public void setUsers(Object users) {
        this.users = users;
    }

    public Object getEditusers() {
        return editusers;
    }

    public void setEditusers(Object editusers) {
        this.editusers = editusers;
    }

    public User getUser(){
        return user;
    }
}
```

```
public void setUser(User user){
    this.user = user;
}

public void setServletRequest(HttpServletRequest request) {
    this.request = request;
}

public HttpServletRequest getServletRequest() {
    return this.request;
}

public String insertUser() throws Exception {
    Logger.debug("UserAction.insertUser()");
    String result = new UserDao().insertUser(user);
    Logger.debug("1 Record Inserted Successfully " + result);
    return ActionSupport.SUCCESS;
}

public String listUsers()
    throws Exception {
    Logger.debug("UserAction.listUsers()");
    users = userDao.listUsers();
    return ActionSupport.SUCCESS;
}

public String deleteUser()
    throws Exception {
    Logger.debug("UserAction.deleteUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    Logger.debug("value="+list.get(1));
    userDao.deleteUser(Integer.parseInt(list.get(1)));
    Logger.debug("1 Record Deleted Successfully ");
    return ActionSupport.SUCCESS;
}

public String editUser()
    throws Exception {
    Logger.debug("UserAction.editUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    editusers=userDao.editUser(Integer.parseInt(list.get(1)));
    return ActionSupport.SUCCESS;
}

public String updateUser()
    throws Exception {
    Logger.debug("UserAction.updateUser()");
    userDao.updateUser(user);
}
```



```
        logger.debug("1 Record Updated Successfully");
        return ActionSupport.SUCCESS;
    }
}
```

iBatisUtil.java:

```
package com.tutorials4u.ibatis;

import java.io.IOException;
import java.io.Reader;
import com.ibatis.common.resources.Resources;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.ibatis.sqlmap.client.SqlMapClientBuilder;

public class IbatisUtil {
    private SqlMapClient sqlmapClient;

    public IbatisUtil() {
    }

    public static SqlMapClient initateSqlMapClient() {
        Reader reader = null;
        try {
            reader = Resources.getResourceAsReader("sql-maps-config.xml");
            SqlMapClient sqlmapClient = SqlMapClientBuilder
                .buildSqlMapClient(reader);
            return sqlmapClient;
        } catch (IOException e) {
            throw new RuntimeException(e.getMessage());
        }
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.ibatis.IbatisUtil;
import com.tutorials4u.model.User;

public class UserDao {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    SqlMapClient sqlmapClient = IbatisUtil.initateSqlMapClient();

    public String insertUser(User user) {
        logger.debug("UserDao.insertUser()");
        try {
            logger.debug("Inserting values....");
            sqlmapClient.insert("user.insertUser", user);
        }
    }
}
```

```
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            return "failure";
        }
        return "success";
    }

    @SuppressWarnings("unchecked")
    public List<User> listUsers() throws SQLException {
        logger.debug("UserDao.listUsers()");
        User rec = new User();
        List<User> user = (List<User>) sqlmapClient.queryForList(
            "user.listUsers", rec);
        return user;
    }

    public void deleteUser(int id) {
        logger.debug("UserDao.deleteUser()");
        try {
            sqlmapClient.delete("user.deleteUser", id);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public User editUser(int id) throws SQLException {
        logger.debug("UserDao.editUser()");
        User user = getUser(id);
        return user;
    }

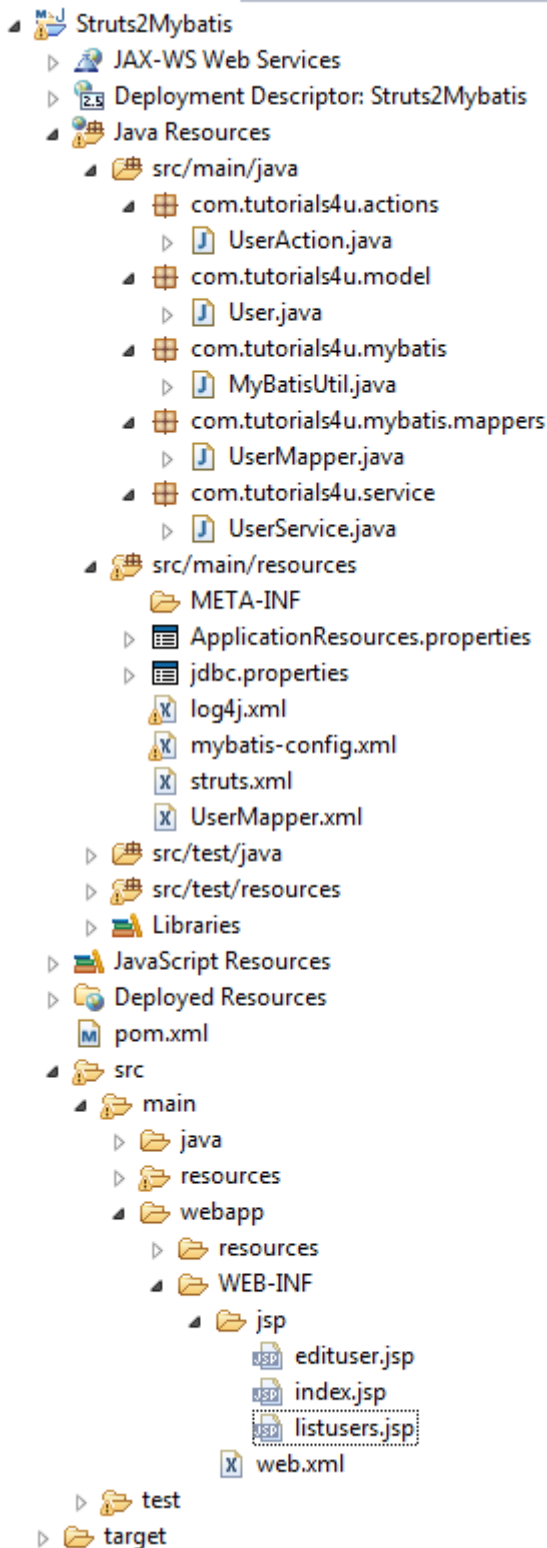
    public User getUser(int id) throws SQLException {
        logger.debug("UserDao.getUser()");
        User user = (User) sqlmapClient.queryForObject("user.getUserById",
id);
        return user;
    }

    public void updateUser(User user) {
        logger.debug("UserDao.updateUser()");
        try {
            sqlmapClient.update("user.updateUser", user);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Index.jsp,edituser.jsp and listUsers.jsp are same as Struts2 JDBC Integration project.

➤ Struts2 MyBatis Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!-- Mybatis framework -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.1.1</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar
    </systemPath>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
    <display-name>Struts2 MyBatis Integration</display-name>
    <filter>
        <filter-name>struts2mybatis</filter-name>
        <filter-class>
            org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
        </filter-class>
    </filter>
    <filter-mapping>
        <filter-name>struts2mybatis</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
</web-app>
```

Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
  <include file="struts-default.xml"/>
  <constant name="struts.enable.DynamicMethodInvocation" value="true" />
  <constant name="struts.devMode" value="false" />
  <constant name="struts.custom.i18n.resources" value="ApplicationResources"
/>
  <package name="default" extends="struts-default">
    <action name="">
      <result>/WEB-INF/jsp/index.jsp</result>
    </action>

    <action name="insertAction" method="insertUser"
class="com.tutorials4u.actions.UserAction">
      <result name="input">/WEB-INF/jsp/index.jsp</result>
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="listUsersAction"
class="com.tutorials4u.actions.UserAction" method="listUsers">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="deleteUserAction"
class="com.tutorials4u.actions.UserAction" method="deleteUser">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

    <action name="editUserAction"
class="com.tutorials4u.actions.UserAction" method="editUser">
      <result name="success">/WEB-INF/jsp/edituser.jsp</result>
    </action>

    <action name="updateUserAction"
class="com.tutorials4u.actions.UserAction" method="updateUser">
      <result name="success">/WEB-INF/jsp/listusers.jsp</result>
    </action>

  </package>
</struts>
```

ApplicationResources.properties:

```
label.firstName = First Name
label.lastName  = Last Name
label.username  = Username
label.password  = Password
label.phone     = Phone
label.emailId   = Email Id
```

jdbc.properties:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

mybatis-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
  <properties resource="jdbc.properties" />
  <typeAliases>
    <typeAlias type="com.tutorials4u.model.User"
      alias="User"></typeAlias>
  </typeAliases>
  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC" />
      <dataSource type="POOLED">
        <property name="driver" value="${jdbc.driverClassName}" />
        <property name="url" value="${jdbc.url}" />
        <property name="username" value="${jdbc.username}" />
        <property name="password" value="${jdbc.password}" />
      </dataSource>
    </environment>
  </environments>
  <mappers>
    <mapper resource="UserMapper.xml" />
  </mappers>
</configuration>
```

UserMapper.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.tutorials4u.mybatis.mappers.UserMapper">
    <resultMap id="UserResultMap" type="User">
        <id property="user_id" column="USER_ID" />
        <result property="username" column="USERNAME"/>
        <result property="password" column="PASSWORD"/>
        <result property="firstName" column="FIRSTNAME"/>
        <result property="lastName" column="LASTNAME"/>
        <result property="phone" column="PHONE"/>
        <result property="emailId" column="EMAILID"/>
    </resultMap>

    <insert id="insertUser" parameterType="user">
        INSERT INTO myuser
        (USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID)
        VALUES
        (#{username},#{password},#{firstName},#{lastName},#{phone},#{emailId})
    </insert>

    <select id="getAllUsers" resultMap="UserResultMap">
        SELECT USER_ID,USERNAME,FIRSTNAME,LASTNAME,PHONE,EMAILID
        FROM myuser;
    </select>

    <select id="getUserById" resultMap="UserResultMap">
        SELECT USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
        FROM myuser
        WHERE USER_ID=#{user_id}
    </select>

    <update id="updateUser" parameterType="user">
        UPDATE myuser
        SET
            USERNAME = #{username},
            PASSWORD = #{password},
            FIRSTNAME= #{firstName},
            LASTNAME = #{lastName},
            PHONE    = #{phone},
            EMAILID  = #{emailId}
        WHERE
            USER_ID = #{user_id}
    </update>

    <delete id="deleteUser" parameterType="String">
        DELETE FROM
            myuser
        WHERE USER_ID = #{user_id}
    </delete>

    <select id="getUserByUsername" resultMap="UserResultMap">
        SELECT USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
        FROM myuser
        WHERE USERNAME = #{username}
    </select>
</mapper>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
    />

        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
public class User{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }
}
```



```
public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.model.User;
import com.tutorials4u.service.UserService;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private UserService userDao = new UserService();
    private User user;
    HttpServletRequest request;

    private Object users, editusers;

    public Object getUsers() {
        return users;
    }

    public void setUsers(Object users) {
        this.users = users;
    }

    public Object getEditusers() {
        return editusers;
    }

    public void setEditusers(Object editusers) {
        this.editusers = editusers;
    }

    public User getUser(){
        return user;
    }

    public void setUser(User user){
        this.user = user;
    }

    public void setServletRequest(HttpServletRequest request) {
        this.request = request;
    }

    public HttpServletRequest getServletRequest() {
        return this.request;
    }

    public String insertUser() throws Exception {
        logger.debug("UserAction.insertUser()");
        new UserService().insertUser(user);
        logger.debug("1 Record Inserted Successfully ");
    }
}
```

```
        return ActionSupport.SUCCESS;
    }

    public String listUsers()
        throws Exception {
        Logger.debug("UserAction.listUsers()");
        users = userDao.getAllUsers();
        return ActionSupport.SUCCESS;
    }

    public String deleteUser()
        throws Exception {
        Logger.debug("UserAction.deleteUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        Logger.debug("value="+list.get(1));
        userDao.deleteUser(Integer.parseInt(list.get(1)));
        Logger.debug("1 Record Deleted Successfully ");
        return ActionSupport.SUCCESS;
    }

    public String editUser()
        throws Exception {
        Logger.debug("UserAction.editUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        editusers=userDao.getUserById(Integer.parseInt(list.get(1)));
        return ActionSupport.SUCCESS;
    }

    public String updateUser()
        throws Exception {
        Logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
        Logger.debug("1 Record Updated Successfully");
        return ActionSupport.SUCCESS;
    }
}
```

MyBatisUtil.java:

```
package com.tutorials4u.mybatis;

import java.io.IOException;
import java.io.Reader;
import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

public class MyBatisUtil {
    private static SqlSessionFactory factory;

    private MyBatisUtil() {
    }

    static {
        Reader reader = null;
        try {
            reader = Resources.getResourceAsReader("mybatis-config.xml");
        } catch (IOException e) {
            throw new RuntimeException(e.getMessage());
        }
        factory = new SqlSessionFactoryBuilder().build(reader);
    }

    public static SqlSessionFactory getSqlSessionFactory() {
        return factory;
    }
}
```

UserMapper.java:

```
package com.tutorials4u.mybatis.mappers;

import java.util.List;
import com.tutorials4u.model.User;

public interface UserMapper {

    public void insertUser(User user);
    public User getUserById(Integer userId);
    public List<User> getAllUsers();
    public void updateUser(User user);
    public void deleteUser(Integer userId);
}
```

UserService.java:

```
package com.tutorials4u.service;

import java.util.List;
import org.apache.ibatis.session.SqlSession;
import com.tutorials4u.model.User;
import com.tutorials4u.mybatis.MyBatisUtil;
import com.tutorials4u.mybatis.mappers.UserMapper;

public class UserService {

    public void insertUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
sqlSession.getMapper(UserMapper.class);
            userMapper.insertUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }

    public User getUserById(Integer userId) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
sqlSession.getMapper(UserMapper.class);
            return userMapper.getUserById(userId);
        } finally {
            sqlSession.close();
        }
    }

    public List<User> getAllUsers() {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
sqlSession.getMapper(UserMapper.class);
            return userMapper.getAllUsers();
        } finally {
            sqlSession.close();
        }
    }

    public void updateUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
sqlSession.getMapper(UserMapper.class);
            userMapper.updateUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }
}
```

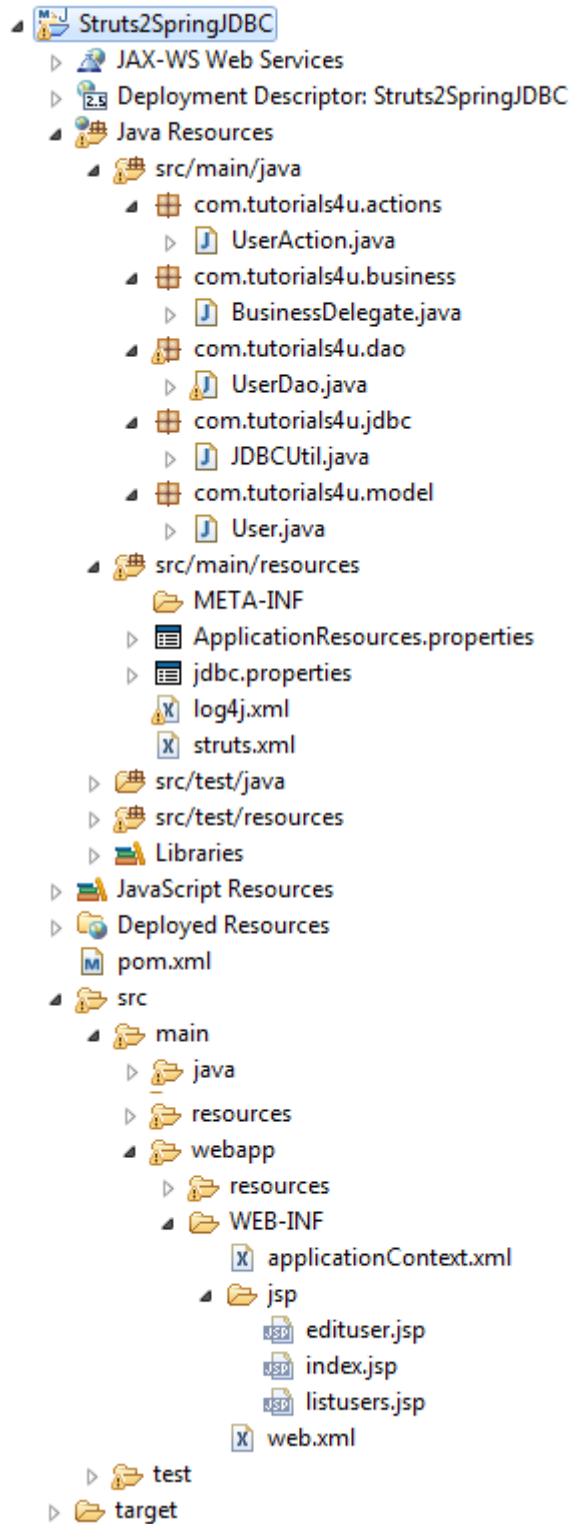
```
        }
    }

    public void deleteUser(Integer userId) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
sqlSession.getMapper(UserMapper.class);
            userMapper.deleteUser(userId);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }
}
```

Index.jsp,edituser.jsp and listUsers.jsp are same as Struts2 JDBC Integration project.

➤ Struts2 Spring JDBC Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-spring-plugin</artifactId>
    <version>2.3.16.1</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar
    </systemPath>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">

    <display-name>Struts2 Spring JDBC Integration</display-name>

    <filter>
        <filter-name>struts2springjdbc</filter-name>
        <filter-class>
            org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
        </filter-class>
    </filter>
    <filter-mapping>
        <filter-name>struts2springjdbc</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
    <listener>
```



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

Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
    <include file="struts-default.xml"/>
    <constant name="struts.enable.DynamicMethodInvocation" value="true" />
    <constant name="struts.devMode" value="false" />
    <constant name="struts.custom.i18n.resources" value="ApplicationResources"
/>
    <package name="default" extends="struts-default">
        <action name="">
            <result>/WEB-INF/jsp/index.jsp</result>
        </action>

        <action name="insertAction" method="insertUser" class="insertAction">
            <result name="input">/WEB-INF/jsp/index.jsp</result>
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="listUsersAction" class="listUsersAction"
method="listUsers">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="deleteUserAction" class="deleteUserAction"
method="deleteUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="editUserAction" class="editUserAction"
method="editUser">
            <result name="success">/WEB-INF/jsp/edituser.jsp</result>
        </action>

        <action name="updateUserAction" class="updateUserAction"
method="updateUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

    </package>
</struts>
```

ApplicationResources.properties:

```
label.firstName = First Name
label.lastName  = Last Name
label.username  = Username
label.password  = Password
label.phone     = Phone
label.emailId   = Email Id
```

jdbc.properties:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.log"
    />
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

applicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <!-- Business Objects -->
    <bean id="businessDelegate"
        class="com.tutorials4u.business.BusinessDelegate">
    </bean>

    <!-- Actions Classes -->
    <bean name="insertAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="listUsersAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="deleteUserAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="editUserAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="updateUserAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>
</beans>
```

Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
    <include file="struts-default.xml"/>
    <constant name="struts.enable.DynamicMethodInvocation" value="true" />
    <constant name="struts.devMode" value="false" />
    <constant name="struts.custom.i18n.resources" value="ApplicationResources"
/>
    <package name="default" extends="struts-default">
        <action name="">
            <result>/WEB-INF/jsp/index.jsp</result>
        </action>

        <action name="insertAction" method="insertUser" class="insertAction">
```

```
<result name="input">/WEB-INF/jsp/index.jsp</result>
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
</action>

<action name="listUsersAction" class="listUsersAction"
method="listUsers">
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
</action>

<action name="deleteUserAction" class="deleteUserAction"
method="deleteUser">
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
</action>

<action name="editUserAction" class="editUserAction"
method="editUser">
    <result name="success">/WEB-INF/jsp/edituser.jsp</result>
</action>

<action name="updateUserAction" class="updateUserAction"
method="updateUser">
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
</action>

</package>
</struts>
```

User.java:

```
package com.tutorials4u.model;
public class User{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }
}
```

```
public void setPassword(String password) {
    this.password = password;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;
import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserDao userDao = new UserDao();

    public String insertUser(User user) throws IOException, SQLException{
        logger.debug("BusinessDelegate.insertUser()");
        String result = new UserDao().insertUser(user);
        return result;
    }

    public List<User> listUsers() throws IOException, SQLException {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.listUsers();
        return users;
    }

    public void deleteUser(int userid) throws IOException, SQLException {
        logger.debug("BusinessDelegate.deleteUser()");
        userDao.deleteUser(userid);
    }

    public User editUser(int userid) throws IOException, SQLException{
        logger.debug("UserAction.editUser()");
        User user = userDao.editUser(userid);
        return user;
    }

    public void updateUser(User user) throws IOException {
        logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
    }
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.tutorials4u.business.BusinessDelegate;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.model.User;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private User user;
    HttpServletRequest request;
    BusinessDelegate businessDelegate;
    private Object users, editusers;

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }

    public void setBusinessDelegate(BusinessDelegate businessDeletage) {
        this.businessDelegate = businessDeletage;
    }

    public Object getUsers() {
        return users;
    }

    public void setUsers(Object users) {
        this.users = users;
    }

    public Object getEditusers() {
        return editusers;
    }

    public void setEditusers(Object editusers) {
        this.editusers = editusers;
    }

    public User getUser(){
        return user;
    }

    public void setUser(User user){
        this.user = user;
    }

    public void setServletRequest(HttpServletRequest request) {
        this.request = request;
    }
}
```

```
public HttpServletRequest getServletRequest() {
    return this.request;
}

public String insertUser() throws Exception {
    logger.debug("UserAction.insertUser()");
    String result = this.businessDelegate.insertUser(user);
    logger.debug("1 Record Inserted Successfully " + result);
    return ActionSupport.SUCCESS;
}

public String listUsers()
    throws Exception {
    logger.debug("UserAction.listUsers()");
    users = this.businessDelegate.listUsers();
    return ActionSupport.SUCCESS;
}

public String deleteUser()
    throws Exception {
    logger.debug("UserAction.deleteUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    logger.debug("value="+list.get(1));
    this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
    logger.debug("1 Record Deleted Successfully ");
    return ActionSupport.SUCCESS;
}

public String editUser()
    throws Exception {
    logger.debug("UserAction.editUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }

    editusers=this.businessDelegate.editUser(Integer.parseInt(list.get(1)));
    return ActionSupport.SUCCESS;
}

public String updateUser()
    throws Exception {
    logger.debug("UserAction.updateUser()");
    this.businessDelegate.updateUser(user);
    logger.debug("1 Record Updated Successfully");
    return ActionSupport.SUCCESS;
}
}
```


JDBCUtil.java:

```
package com.tutorials4u.jdbc;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.util.Properties;

public class JDBCUtil {
    private static Connection connection = null;
    InputStream inputStream;
    String propFileName = "jdbc.properties";
    public static Properties prop = new Properties();
    public JDBCUtil() throws IOException {
        inputStream = getClass().getClassLoader().getResourceAsStream(
            propFileName);
        if (inputStream != null) {
            prop.load(inputStream);
        } else {
            throw new FileNotFoundException("property file '" +
propFileName
                                + "' not found in the classpath");
        }
    }
    public static Connection getMysqlConnection() throws IOException {
        String driverClass = prop.getProperty("jdbc.driverClassName");
        String url = prop.getProperty("jdbc.url");
        String username = prop.getProperty("jdbc.username");
        String password = prop.getProperty("jdbc.password");

        if (connection == null) {
            try {
                Class.forName(driverClass).newInstance();
                connection = DriverManager.getConnection(url, username,
                    password);
                System.out.println("Connected To Mysql Datasbase");
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
        return connection;
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;

import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.jdbc.JDBCUtil;
import com.tutorials4u.model.User;

public class UserDao {
    private static final Logger logger = Logger.getLogger(UserAction.class);

    PreparedStatement stmt = null;

    @SuppressWarnings("static-access")
    public String insertUser(User newuser) throws IOException, SQLException {
        logger.debug("UserDao.createUser()");
        JDBCUtil util = new JDBCUtil();
        Connection conn = util.getMysqlConnection();
        try {
            String username = newuser.getUsername();
            String password = newuser.getPassword();
            String firstname = newuser.getFirstName();
            String lastname = newuser.getLastName();
            String emailId = newuser.getEmailId();
            String phone = newuser.getPhone();

            String sql = new String(
                "insert into\n"
                + "myuser(USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE\n"
                + ",EMAILID) values(?,?,?,?,?,?)");

            stmt = conn.prepareStatement(sql);
            stmt.setString(1, username);
            stmt.setString(2, password);
            stmt.setString(3, firstname);
            stmt.setString(4, lastname);
            stmt.setString(5, phone);
            stmt.setString(6, emailId);

            logger.debug("Inserting values....");
            stmt.executeUpdate();
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            return "failure";
        } finally {
            stmt.close();
            conn.close();
        }
        return "success";
    }
}
```

```
@SuppressWarnings("static-access")
public List<User> listUsers() throws IOException, SQLException {
    Logger.debug("UserDao.listUsers()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    List<User> users = new ArrayList<User>();
    try {
        String sql = "select
USER_ID,USERNAME,FIRSTNAME,LASTNAME,PHONE,EMAILID from
myuser";
        statement = conn.createStatement();
        result = statement.executeQuery(sql);
        User newuser = null;
        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
            users.add(newuser);
        }
    } catch (SQLException e) {
        Logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
        // conn.close();
    }
    return users;
}

@SuppressWarnings("static-access")
public void deleteUser(int id) throws IOException, SQLException {
    Logger.debug("UserDao.deleteUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    Statement statement = null;
    try {
        String sql = "delete from myuser where user_id=" + id;
        statement = conn.createStatement();
        statement.executeUpdate(sql);
    } catch (SQLException e) {
        Logger.error("Error deleting User :" + e.getMessage());
    } finally {
        statement.close();
        // conn.close();
    }
}

@SuppressWarnings("static-access")
public User editUser(int id) throws IOException, SQLException {
    Logger.debug("UserDao.editUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
}
```

```
User user = getUser(id);
return user;
}

@SuppressWarnings("static-access")
public User getUser(int id) throws SQLException, IOException {
    Logger.debug("UserDao.getUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    ResultSet result = null;
    Statement statement = null;
    User newuser = null;
    try {
        String sql = "select
USER_ID,USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID
from myuser WHERE USER_ID="
+ id;
        statement = conn.createStatement();
        result = statement.executeQuery(sql);

        while (result.next()) {
            newuser = new User();
            newuser.setUser_id(result.getInt("USER_ID"));
            newuser.setUsername(result.getString("USERNAME"));
            newuser.setPassword(result.getString("PASSWORD"));
            newuser.setFirstName(result.getString("FIRSTNAME"));
            newuser.setLastName(result.getString("LASTNAME"));
            newuser.setPhone(result.getString("PHONE"));
            newuser.setEmailId(result.getString("EMAILID"));
        }
    } catch (SQLException e) {
        Logger.error("Error listing Users :" + e.getMessage());
    } finally {
        result.close();
        statement.close();
        // conn.close();
    }
    return newuser;
}

@SuppressWarnings("static-access")
public void updateUser(User newuser) throws IOException {
    Logger.debug("UserDao.updateUser()");
    JDBCUtil util = new JDBCUtil();
    Connection conn = util.getMysqlConnection();
    try {
        int userid = newuser.getUser_id();
        String username = newuser.getUsername();
        String password = newuser.getPassword();
        String firstname = newuser.getFirstName();
        String lastname = newuser.getLastName();
        String emailId = newuser.getEmailId();
        String phone = newuser.getPhone();

        String sql = "UPDATE MYUSER SET
USERNAME=?,PASSWORD=?,FIRSTNAME=?,LASTNAME=?,PHONE=?,EMAILID=?
WHERE USER_ID="+userid;

        stmt = conn.prepareStatement(sql);
```














































```
        stmt.setString(1, username);
        stmt.setString(2, password);
        stmt.setString(3, firstname);
        stmt.setString(4, lastname);
        stmt.setString(5, phone);
        stmt.setString(6, emailId);

        logger.debug("Updating values....");
        stmt.executeUpdate();
    } catch (Exception e) {
        logger.error("Error updating User :" + e.getMessage());
    }
}
}
```

Index.jsp,edituser.jsp and listUsers.jsp are same as Struts2 JDBC Integration project.

➤ Struts2 Spring Hibernate Integration

Project Structure:

- ▲  Struts2SpringHibernate
 - ▷  JAX-WS Web Services
 - ▷  Deployment Descriptor: Struts2SpringHibernate
 - ▲  Java Resources
 - ▲  src/main/java
 - ▲  com.tutorials4u.actions
 - ▷  UserAction.java
 - ▲  com.tutorials4u.business
 - ▷  BusinessDelegate.java
 - ▲  com.tutorials4u.dao
 - ▷  UserDao.java
 - ▲  com.tutorials4u.hibernate
 - ▷  HibernateUtil.java
 - ▲  com.tutorials4u.model
 - ▷  User.java
 - ▲  src/main/resources
 - ▲  META-INF
 - ▷  ApplicationResources.properties
 - ▷  hibernate.cfg.xml
 - ▷  jdbc.properties
 - ▷  log4j.xml
 - ▷  struts.xml
 - ▷  User.hbm.xml
 - ▷  src/test/java
 - ▷  src/test/resources
 - ▷  Libraries
 - ▷  JavaScript Resources
 - ▷  Deployed Resources
 - ▷  pom.xml
 - ▲  src
 - ▲  main
 - ▷  java
 - ▷  resources
 - ▲  webapp
 - ▷  resources
 - ▲  WEB-INF
 - ▷  applicationContext.xml
 - ▲  jsp
 - ▷  edituser.jsp
 - ▷  index.jsp
 - ▷  listusers.jsp
 - ▷  success.jsp
 - ▷  web.xml
 - ▷  test
 - ▷  target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-spring-plugin</artifactId>
    <version>2.3.16.1</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar
    </systemPath>
</dependency>

<!-- Hibernate framework -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-core</artifactId>
    <version>4.2.4.Final</version>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
  <display-name>Struts2 Spring Hibernate Integration </display-name>
  <filter>
    <filter-name>struts2springhibernate</filter-name>
    <filter-class>
      org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
    </filter-class>
  </filter>
  <filter-mapping>
    <filter-name>struts2springhibernate</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <listener>
    <listener-class>
      org.springframework.web.context.ContextLoaderListener
    </listener-class>
  </listener>
</web-app>
```

ApplicationResources.properties:

```
label.firstName = First Name
label.lastName  = Last Name
label.username  = Username
label.password  = Password
label.phone     = Phone
label.emailId   = Email Id
```

jdbc.properties:

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/hibernate
jdbc.username=root
jdbc.password=
```

hibernate.cfg.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
  <session-factory>
    <property
name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
    <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/hibernate</property>
    <property name="hibernate.connection.username">root</property>
    <property name="hibernate.connection.password"></property>
    <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
    <property name="show_sql">true</property>
    <property name="format_sql">true</property>
    <property name="hbm2ddl.auto">update</property>
```



```
        <mapping resource="User.hbm.xml" />
    </session-factory>
</hibernate-configuration>
```

User.hbm.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD//EN"
    "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.tutorials4u.model.User" table="MYUSER">
        <meta attribute="class-description">
            This class contains the User detail.
        </meta>
        <id name="user_id" type="int" column="USER_ID">
            <generator class="native" />
        </id>
        <property name="username" column="USERNAME" type="string" />
        <property name="password" column="PASSWORD" type="string" />
        <property name="firstName" column="FIRSTNAME" type="string" />
        <property name="lastName" column="LASTNAME" type="string" />
        <property name="phone" column="PHONE" type="string" />
        <property name="emailId" column="EMAILID" type="string" />
    </class>
</hibernate-mapping>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.log"
    />

        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
    </root>
</log4j:configuration>
```

```
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

applicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <!-- Business Objects -->
    <bean id="businessDelegate"
        class="com.tutorials4u.business.BusinessDelegate">
    </bean>

    <!-- Actions Classes -->
    <bean name="insertAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="listUsersAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="deleteUserAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="editUserAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>

    <bean name="updateUserAction" class="com.tutorials4u.actions.UserAction">
        <property name="businessDelegate"><ref
local="businessDelegate"/></property>
    </bean>
</beans>
```

Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
    <include file="struts-default.xml"/>

    <constant name="struts.enable.DynamicMethodInvocation" value="true" />
    <constant name="struts.devMode" value="false" />
    <constant name="struts.custom.i18n.resources" value="ApplicationResources"
/>
```

```
<package name="default" extends="struts-default">
  <action name="">
    <result>/WEB-INF/jsp/index.jsp</result>
  </action>

  <action name="insertAction" method="insertUser"
class="com.tutorials4u.actions.UserAction">
    <result name="input">/WEB-INF/jsp/index.jsp</result>
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
  </action>

  <action name="listUsersAction"
class="com.tutorials4u.actions.UserAction" method="listUsers">
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
  </action>

  <action name="deleteUserAction"
class="com.tutorials4u.actions.UserAction" method="deleteUser">
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
  </action>

  <action name="editUserAction"
class="com.tutorials4u.actions.UserAction" method="editUser">
    <result name="success">/WEB-INF/jsp/edituser.jsp</result>
  </action>

  <action name="updateUserAction"
class="com.tutorials4u.actions.UserAction" method="updateUser">
    <result name="success">/WEB-INF/jsp/listusers.jsp</result>
  </action>

</package>
</struts>
```

User.java:

```
package com.tutorials4u.model;
public class User{
  private int user_id;
  String username;
  String password;
  String firstName;
  String lastName;
  String phone;
  String emailId;

  public int getUser_id() {
    return user_id;
  }

  public void setUser_id(int user_id) {
    this.user_id = user_id;
  }

  public String getUsername() {
    return username;
  }
}
```

```
public void setUsername(String username) {
    this.username = username;
}

public String getPassword() {
    return password;
}

public void setPassword(String password) {
    this.password = password;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getPhone() {
    return phone;
}

public void setPhone(String phone) {
    this.phone = phone;
}

public String getEmailId() {
    return emailId;
}

public void setEmailId(String emailId) {
    this.emailId = emailId;
}

@Override
public String toString() {
    return "Id: " + user_id + "; fname : " + firstName + "' phone : "
        + phone + "; emailId: " + emailId;
}
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.tutorials4u.business.BusinessDelegate;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.model.User;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private User user;
    HttpServletRequest request;
    BusinessDelegate businessDelegate;
    private Object users, editusers;

    public Object getUsers() {
        return users;
    }

    public void setUsers(Object users) {
        this.users = users;
    }

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }

    public void setBusinessDelegate(BusinessDelegate businessDeletage) {
        this.businessDelegate = businessDeletage;
    }

    public Object getEditusers() {
        return editusers;
    }

    public void setEditusers(Object editusers) {
        this.editusers = editusers;
    }

    public User getUser(){
        return user;
    }

    public void setUser(User user){
        this.user = user;
    }

    public void setServletRequest(HttpServletRequest request) {
        this.request = request;
    }

    public HttpServletRequest getServletRequest() {
```

```
        return this.request;
    }

    public String insertUser() throws Exception {
        logger.debug("UserAction.insertUser()");
        String result = this.businessDelegate.insertUser(user);
        logger.debug("1 Record Inserted Successfully " + result);
        return ActionSupport.SUCCESS;
    }

    public String listUsers()
        throws Exception {
        logger.debug("UserAction.listUsers()");
        users = this.businessDelegate.listUsers();
        return ActionSupport.SUCCESS;
    }

    public String deleteUser()
        throws Exception {
        logger.debug("UserAction.deleteUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }
        logger.debug("value="+list.get(1));
        this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
        logger.debug("1 Record Deleted Successfully ");
        return ActionSupport.SUCCESS;
    }

    public String editUser()
        throws Exception {
        logger.debug("UserAction.editUser()");
        String userid=request.getQueryString();
        ArrayList<String> list=new ArrayList<String>();
        for(String newstr:userid.split("=")) {
            list.add(newstr);
        }

        editusers=this.businessDelegate.editUser(Integer.parseInt(list.get(1)));
        return ActionSupport.SUCCESS;
    }

    public String updateUser()
        throws Exception {
        logger.debug("UserAction.updateUser()");
        this.businessDelegate.updateUser(user);
        logger.debug("1 Record Updated Successfully");
        return ActionSupport.SUCCESS;
    }
}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;
import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserDao userDao = new UserDao();

    public String insertUser(User user) throws IOException, SQLException{
        logger.debug("BusinessDelegate.insertUser()");
        String result = new UserDao().insertUser(user);
        return result;
    }

    public List<User> listUsers() throws IOException, SQLException {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.listUsers();
        return users;
    }

    public void deleteUser(int userid) throws IOException, SQLException {
        logger.debug("BusinessDelegate.deleteUser()");
        userDao.deleteUser(userid);
    }

    public User editUser(int userid) throws IOException, SQLException{
        logger.debug("UserAction.editUser()");
        User user = userDao.editUser(userid);
        return user;
    }

    public void updateUser(User user) throws IOException {
        logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
    }
}
```

HibernateUtil.java:

```
package com.tutorials4u.hibernate;

import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class HibernateUtil {

    private static final SessionFactory sessionFactory =
        buildSessionFactory();

    private static SessionFactory buildSessionFactory() {
        try {
            // Create the SessionFactory from hibernate.cfg.xml
            return new Configuration().configure().buildSessionFactory();
        }
        catch (Throwable ex) {
            // Make sure you log the exception, as it might be swallowed
            System.err.println("Initial SessionFactory creation failed." +
ex);

            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

    public static void shutdown() {
        // Close caches and connection pools
        getSessionFactory().close();
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;

import java.util.List;
import org.hibernate.Session;
import org.apache.log4j.Logger;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.hibernate.HibernateUtil;
import com.tutorials4u.model.User;

public class UserDao {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);

    public String insertUser(User newuser) {
        logger.debug("UserDao.createUser()");
        Session session = HibernateUtil.getSessionFactory().openSession();
        try {
            session.beginTransaction();
            User user = new User();
            user.setUsername(newuser.getUsername());
            user.setPassword(newuser.getPassword());
            user.setFirstName(newuser.getFirstName());
        }
```



```
        user.setLastName(newuser.getLastName());
        user.setEmailId(newuser.getEmailId());
        user.setPhone(newuser.getPhone());
        logger.debug("Inserting values...");
        session.save(user);
        session.getTransaction().commit();
    } catch (Exception e) {
        logger.error("Error creating User :" + e.getMessage());
        session.getTransaction().rollback();
        return "failure";
    }
    return "success";
}

@SuppressWarnings("unchecked")
public List<User> listUsers() {
    logger.debug("UserDao.listUsers()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    return session.createQuery("From User").list();
}

public void deleteUser(int id) {
    logger.debug("UserDao.deleteUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    User user = getUser(id);
    session.delete(user);
    session.getTransaction().commit();
}

public User editUser(int id) {
    logger.debug("UserDao.editUser()");
    @SuppressWarnings("unused")
    Session session = HibernateUtil.getSessionFactory().openSession();
    User user = getUser(id);
    return user;
}

public User getUser(int id) {
    logger.debug("UserDao.getUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    User user = (User) session.get(User.class, id);
    return user;
}

public void updateUser(User user) {
    logger.debug("UserDao.updateUser()");
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.update(user);
    session.getTransaction().commit();
}
}
```

Index.jsp,edituser.jsp and listUsers.jsp are same as Struts2 JDBC Integration project.

➤ Struts2 Spring iBatis Integration

Project Structure:

- ▲ Struts2SpringIbatis
 - ▷ JAX-WS Web Services
 - ▷ Deployment Descriptor: Struts2SpringIbatis
 - ▲ Java Resources
 - ▲ src/main/java
 - ▲ com.tutorials4u.actions
 - ▷ UserAction.java
 - ▲ com.tutorials4u.business
 - ▷ BusinessDelegate.java
 - ▲ com.tutorials4u.dao
 - ▷ UserDao.java
 - ▲ com.tutorials4u.ibatis
 - ▷ IbatisUtil.java
 - ▲ com.tutorials4u.model
 - ▷ User.java
 - ▲ src/main/resources
 - ▲ META-INF
 - ▷ ApplicationResources.properties
 - ▷ jdbc.properties
 - ▷ log4j.xml
 - ▷ sql-maps-config.xml
 - ▷ struts.xml
 - ▷ user.xml
 - ▷ src/test/java
 - ▷ src/test/resources
 - ▷ Libraries
 - ▷ JavaScript Resources
 - ▷ Deployed Resources
 - ▷ pom.xml
 - ▲ src
 - ▲ main
 - ▷ java
 - ▷ resources
 - ▲ webapp
 - ▷ resources
 - ▲ WEB-INF
 - ▷ applicationContext.xml
 - ▲ jsp
 - ▷ edituser.jsp
 - ▷ index.jsp
 - ▷ listusers.jsp
 - ▷ web.xml
 - ▷ test
 - ▷ target

Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-spring-plugin</artifactId>
    <version>2.3.16.1</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar
    </systemPath>
</dependency>

<!-- Ibatis -->
<dependency>
    <groupId>org.apache.ibatis</groupId>
    <artifactId>ibatis-sqlmap</artifactId>
    <version>2.3.4.726</version>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
    <display-name>Struts2 + Spring + iBatis Integration</display-name>
    <filter>
        <filter-name>struts2springibatis</filter-name>
        <filter-class>
            org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
        </filter-class>
    </filter>
```

```
<filter-mapping>
  <filter-name>struts2springibatis</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
  <listener>
    <listener-class>
      org.springframework.web.context.ContextLoaderListener
    </listener-class>
  </listener>
</web-app>
```

ApplicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
  <!-- Business Objects -->
  <bean id="businessDelegate"
    class="com.tutorials4u.business.BusinessDelegate">
  </bean>

  <!-- Actions Classes -->
  <bean name="insertAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="listUsersAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="deleteUserAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="editUserAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="updateUserAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>
</beans>
```

ApplicationResources.properties:

```
label.firstName = First Name
label.lastName  = Last Name
label.username  = Username
label.password  = Password
label.phone     = Phone
label.emailId   = Email Id
```

Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
    <include file="struts-default.xml"/>
    <constant name="struts.enable.DynamicMethodInvocation" value="true" />
    <constant name="struts.devMode" value="false" />
    <constant name="struts.custom.i18n.resources" value="ApplicationResources" />
    </>
    <package name="default" extends="struts-default">
        <action name="">
            <result>/WEB-INF/jsp/index.jsp</result>
        </action>

        <action name="insertAction" method="insertUser"
class="com.tutorials4u.actions.UserAction">
            <result name="input">/WEB-INF/jsp/index.jsp</result>
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="listUsersAction"
class="com.tutorials4u.actions.UserAction" method="listUsers">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="deleteUserAction"
class="com.tutorials4u.actions.UserAction" method="deleteUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="editUserAction"
class="com.tutorials4u.actions.UserAction" method="editUser">
            <result name="success">/WEB-INF/jsp/edituser.jsp</result>
        </action>

        <action name="updateUserAction"
class="com.tutorials4u.actions.UserAction" method="updateUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

    </package>
</struts>
```

sql-maps-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE sqlMapConfig
    PUBLIC "-//ibatis.apache.org//DTD SQL Map Config 2.0//EN"
    "http://ibatis.apache.org/dtd/sql-map-config-2.dtd">
<sqlMapConfig>

    <settings useStatementNamespaces="true"/>

    <transactionManager type="JDBC">
        <dataSource type="SIMPLE">
            <property name="JDBC.Driver" value="com.mysql.jdbc.Driver"/>
            <property name="JDBC.ConnectionURL"
value="jdbc:mysql://localhost:3306/hibernate"/>
            <property name="JDBC.Username" value="root"/>
            <property name="JDBC.Password" value=""/>
        </dataSource>
    </transactionManager>

    <sqlMap resource="user.xml"/>

</sqlMapConfig>
```

User.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sqlMap PUBLIC "-//ibatis.apache.org//DTD SQL Map 2.0//EN"
"http://ibatis.apache.org/dtd/sql-map-2.dtd">
<sqlMap namespace="user">

    <typeAlias alias="USER" type="com.tutorials4u.model.User" />

    <resultMap id="userResultMap" class="USER">
        <result property="user_id" column="USER_ID" />
        <result property="username" column="USERNAME" />
        <result property="password" column="PASSWORD" />
        <result property="firstName" column="FIRSTNAME" />
        <result property="lastName" column="LASTNAME" />
        <result property="phone" column="PHONE" />
        <result property="emailId" column="EMAILID" />
    </resultMap>

    <select id="listUsers" resultMap="userResultMap">
        SELECT * FROM MYUSER
    </select>

    <insert id="insertUser" parameterClass="USER">
        INSERT INTO MYUSER
        (USERNAME,PASSWORD,FIRSTNAME,LASTNAME,PHONE,EMAILID)

        VALUES(#username#,#password#,#firstName#,#lastName#,#phone#,#emailId#);
    </insert>

    <delete id="deleteUser" parameterClass="java.Lang.Integer">
        DELETE FROM MYUSER WHERE USER_ID = #value#
    </delete>
```

```
<select id="getUserById" parameterClass="java.Lang.Integer"
        resultMap="userResultMap">
    SELECT * FROM MYUSER WHERE USER_ID = #value#
</select>

<update id="updateUser" parameterClass="USER">
    UPDATE MYUSER
    SET USERNAME = #username#,
    PASSWORD     = #password#,
    FIRSTNAME    = #firstName#,
    LASTNAME     = #lastName#,
    PHONE        = #phone#,
    EMAILID      = #emailId#
    WHERE USER_ID = #user_id#
</update>
```

```
</sqlMap>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
    />

        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
public class User{
    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }
}
```



```
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }

    @Override
    public String toString() {
        return "Id: " + user_id + "; fname : " + firstName + "' phone : "
            + phone + "; emailId: " + emailId;
    }
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;

import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.tutorials4u.business.BusinessDelegate;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.model.User;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private User user;
    HttpServletRequest request;
    BusinessDelegate businessDelegate;
    private Object users, editusers;

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }

    public void setBusinessDelegate(BusinessDelegate businessDeletage) {
        this.businessDelegate = businessDeletage;
    }

    public Object getUsers() {
        return users;
    }

    public void setUsers(Object users) {
        this.users = users;
    }

    public Object getEditusers() {
        return editusers;
    }
}
```

```
public void setEditusers(Object editusers) {
    this.editusers = editusers;
}

public User getUser(){
    return user;
}

public void setUser(User user){
    this.user = user;
}

public void setServletRequest(HttpServletRequest request) {
    this.request = request;
}

public HttpServletRequest getServletRequest() {
    return this.request;
}

public String insertUser() throws Exception {
    logger.debug("UserAction.insertUser()");
    String result = this.businessDelegate.insertUser(user);
    logger.debug("1 Record Inserted Successfully " + result);
    return ActionSupport.SUCCESS;
}

public String listUsers()
    throws Exception {
    logger.debug("UserAction.listUsers()");
    users = this.businessDelegate.listUsers();
    return ActionSupport.SUCCESS;
}

public String deleteUser()
    throws Exception {
    logger.debug("UserAction.deleteUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
    logger.debug("value="+list.get(1));
    this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
    logger.debug("1 Record Deleted Successfully ");
    return ActionSupport.SUCCESS;
}

public String editUser()
    throws Exception {
    logger.debug("UserAction.editUser()");
    String userid=request.getQueryString();
    ArrayList<String> list=new ArrayList<String>();
    for(String newstr:userid.split("=")) {
        list.add(newstr);
    }
}
```

```
        editusers=this.businessDelegate.editUser(Integer.parseInt(list.get(1)));
        return ActionSupport.SUCCESS;
    }

    public String updateUser()
        throws Exception {
        logger.debug("UserAction.updateUser()");
        this.businessDelegate.updateUser(user);
        logger.debug("1 Record Updated Successfully");
        return ActionSupport.SUCCESS;
    }
}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;
import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserDao userDao = new UserDao();

    public String insertUser(User user) throws IOException, SQLException{
        logger.debug("BusinessDelegate.insertUser()");
        String result = new UserDao().insertUser(user);
        return result;
    }

    public List<User> listUsers() throws IOException, SQLException {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.listUsers();
        return users;
    }

    public void deleteUser(int userid) throws IOException, SQLException {
        logger.debug("BusinessDelegate.deleteUser()");
        userDao.deleteUser(userid);
    }

    public User editUser(int userid) throws IOException, SQLException{
        logger.debug("UserAction.editUser()");
        User user = userDao.editUser(userid);
        return user;
    }

    public void updateUser(User user) throws IOException {
        logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
    }
}
```

IbatisUtil.java:

```
package com.tutorials4u.ibatis;

import java.io.IOException;
import java.io.Reader;
import com.ibatis.common.resources.Resources;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.ibatis.sqlmap.client.SqlMapClientBuilder;

public class IbatisUtil {
    private SqlMapClient sqlmapClient;

    public IbatisUtil() {
    }

    public static SqlMapClient initateSqlMapClient() {
        Reader reader = null;
        try {
            reader = Resources.getResourceAsReader("sql-maps-config.xml");
            SqlMapClient sqlmapClient = SqlMapClientBuilder
                .buildSqlMapClient(reader);
            return sqlmapClient;
        } catch (IOException e) {
            throw new RuntimeException(e.getMessage());
        }
    }
}
```

UserDao.java:

```
package com.tutorials4u.dao;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.tutorials4u.actions.UserAction;
import com.tutorials4u.ibatis.IbatisUtil;
import com.tutorials4u.model.User;

public class UserDao {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    SqlMapClient sqlmapClient = IbatisUtil.initateSqlMapClient();

    public String insertUser(User user) {
        logger.debug("UserDao.insertUser()");
        try {
            logger.debug("Inserting values...");
            sqlmapClient.insert("user.insertUser", user);
        } catch (Exception e) {
            logger.error("Error creating User :" + e.getMessage());
            return "failure";
        }
        return "success";
    }

    @SuppressWarnings("unchecked")
```

```
public List<User> listUsers() throws SQLException {
    logger.debug("UserDao.listUsers()");
    User rec = new User();
    List<User> user = (List<User>) sqlmapClient.queryForList(
        "user.listUsers", rec);
    return user;
}

public void deleteUser(int id) {
    logger.debug("UserDao.deleteUser()");
    try {
        sqlmapClient.delete("user.deleteUser", id);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public User editUser(int id) throws SQLException {
    logger.debug("UserDao.editUser()");
    User user = getUser(id);
    return user;
}

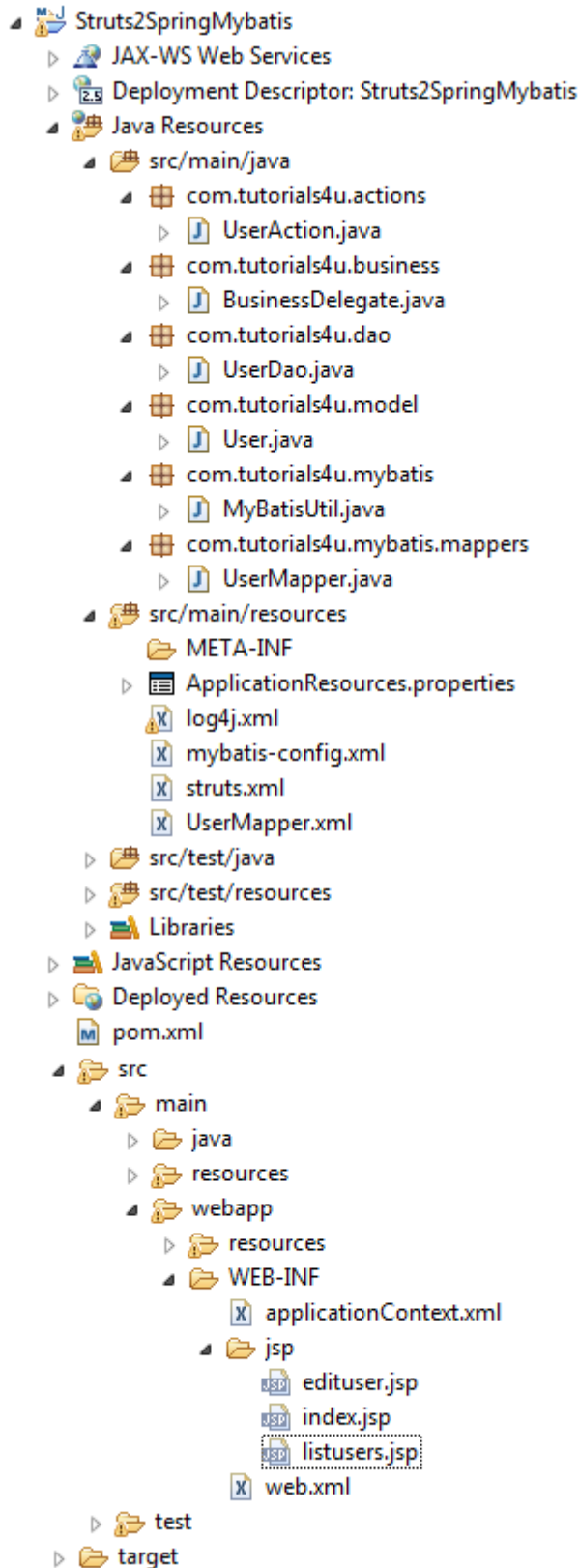
public User getUser(int id) throws SQLException {
    logger.debug("UserDao.getUser()");
    User user = (User) sqlmapClient.queryForObject("user.getUserById",
id);
    return user;
}

public void updateUser(User user) {
    logger.debug("UserDao.updateUser()");
    try {
        sqlmapClient.update("user.updateUser", user);
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}
```

Index.jsp,edituser.jsp and listUsers.jsp are same as Struts2 JDBC Integration project.

➤ Struts2 Spring MyBatis Integration

Project Structure:



Pom.xml:

```
<!-- Struts -->
<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-core</artifactId>
    <version>2.3.16.3</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<dependency>
    <groupId>org.apache.struts</groupId>
    <artifactId>struts2-spring-plugin</artifactId>
    <version>2.3.16.1</version>
</dependency>

<dependency>
    <groupId>com.sun</groupId>
    <artifactId>tools</artifactId>
    <version>1.6</version>
    <scope>system</scope>
    <systemPath>
        C:\Program Files\Java\jdk1.7.0\lib\tools.jar
    </systemPath>
</dependency>

<!-- Mybatis framework -->
<dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.1.1</version>
</dependency>
```

Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:web="http://xmlns.jcp.org/xml/ns/javaee">
  <display-name>Struts2 + Spring + MyBatis Integration</display-name>
  <filter>
    <filter-name>struts2springmybatis</filter-name>
    <filter-
class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-
r-class>
  </filter>
  <filter-mapping>
    <filter-name>struts2springmybatis</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <listener>
    <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>
</web-app>
```

ApplicationContext.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN"
"http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
  <!-- Business Objects -->
  <bean id="businessDelegate"
        class="com.tutorials4u.business.BusinessDelegate">
  </bean>

  <!-- Actions Classes -->
  <bean name="insertAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="listUsersAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="deleteUserAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="editUserAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>

  <bean name="updateUserAction" class="com.tutorials4u.actions.UserAction">
    <property name="businessDelegate"><ref
local="businessDelegate"/></property>
  </bean>
</beans>
```


Struts.xml:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
    <include file="struts-default.xml"/>

    <constant name="struts.enable.DynamicMethodInvocation" value="true" />
    <constant name="struts.devMode" value="false" />
    <constant name="struts.custom.i18n.resources" value="ApplicationResources"
/>

    <package name="default" extends="struts-default">
        <action name="">
            <result>/WEB-INF/jsp/index.jsp</result>
        </action>

        <action name="insertAction" method="insertUser"
class="com.tutorials4u.actions.UserAction">
            <result name="input">/WEB-INF/jsp/index.jsp</result>
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="listUsersAction"
class="com.tutorials4u.actions.UserAction" method="listUsers">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="deleteUserAction"
class="com.tutorials4u.actions.UserAction" method="deleteUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

        <action name="editUserAction"
class="com.tutorials4u.actions.UserAction" method="editUser">
            <result name="success">/WEB-INF/jsp/edituser.jsp</result>
        </action>

        <action name="updateUserAction"
class="com.tutorials4u.actions.UserAction" method="updateUser">
            <result name="success">/WEB-INF/jsp/listusers.jsp</result>
        </action>

    </package>
</struts>
```

ApplicationResources.properties:

```
label.firstName = First Name
label.lastName  = Last Name
label.username  = Username
label.password  = Password
label.phone     = Phone
label.emailId   = Email Id
```

mybatis-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
  <properties resource="jdbc.properties" />
  <typeAliases>
    <typeAlias type="com.tutorials4u.model.User"
      alias="User"></typeAlias>
  </typeAliases>
  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC" />
      <dataSource type="POOLED">
        <property name="driver" value="${jdbc.driverClassName}" />
        <property name="url" value="${jdbc.url}" />
        <property name="username" value="${jdbc.username}" />
        <property name="password" value="${jdbc.password}" />
      </dataSource>
    </environment>
  </environments>
  <mappers>
    <mapper resource="UserMapper.xml" />
  </mappers>
</configuration>
```

UserMapper.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.tutorials4u.mybatis.mappers.UserMapper">

  <resultMap id="UserResultMap" type="User">
    <id property="user_id" column="USER_ID" />
    <result property="username" column="USERNAME"/>
    <result property="password" column="PASSWORD"/>
    <result property="firstName" column="FIRSTNAME"/>
    <result property="lastName" column="LASTNAME"/>
    <result property="phone" column="PHONE"/>
    <result property="emailId" column="EMAILID"/>
  </resultMap>

  <insert id="insertUser" parameterType="user">
    INSERT INTO myuser
    (USERNAME,PASSWORD,FIRSTNAME, LASTNAME, PHONE, EMAILID)
    VALUES
    (#{username},#{password},#{firstName},#{lastName},#{phone},#{emailId})
  </insert>

  <select id="getAllUsers" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,FIRSTNAME, LASTNAME, PHONE, EMAILID
    FROM myuser;
  </select>

  <select id="getUserById" resultMap="UserResultMap">
    SELECT USER_ID,USERNAME,PASSWORD,FIRSTNAME, LASTNAME, PHONE, EMAILID
```

```
        FROM myuser
        WHERE USER_ID=#{user_id}
    </select>

    <update id="updateUser" parameterType="user">
        UPDATE myuser
    SET
        USERNAME = #{username},
        PASSWORD = #{password},
        FIRSTNAME= #{firstName},
        LASTNAME = #{lastName},
        PHONE    = #{phone},
        EMAILID  = #{emailId}
    WHERE
        USER_ID = #{user_id}
    </update>

    <delete id="deleteUser" parameterType="String">
        DELETE FROM
            myuser
        WHERE USER_ID = #{user_id}
    </delete>

    <select id="getUserByUsername" resultMap="UserResultMap">
        SELECT USER_ID, USERNAME, PASSWORD, FIRSTNAME, LASTNAME, PHONE, EMAILID
        FROM myuser
        WHERE USERNAME = #{username}
    </select>
</mapper>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/StrutsCoding/Struts1.x/Logs/Logs.Log"
    />

        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>
```

```
<root>
  <level value="DEBUG" />
  <appender-ref ref="console" />
  <appender-ref ref="file" />
</root>
</log4j:configuration>
```

User.java:

```
package com.tutorials4u.model;
public class User{

    private int user_id;
    String username;
    String password;
    String firstName;
    String lastName;
    String phone;
    String emailId;

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
```

```
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }

    @Override
    public String toString() {
        return "Id: " + user_id + "; fname : " + firstName + "' phone : "
            + phone + "; emailId: " + emailId;
    }
}
```

UserAction.java:

```
package com.tutorials4u.actions;

import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.ServletRequestAware;
import com.tutorials4u.business.BusinessDelegate;
import com.opensymphony.xwork2.ActionSupport;
import com.tutorials4u.model.User;

@SuppressWarnings("serial")
public class UserAction extends ActionSupport implements ServletRequestAware {
    // Get a logger
    private static final Logger logger = Logger.getLogger(UserAction.class);
    private User user;
    HttpServletRequest request;
    BusinessDelegate businessDelegate;
    private Object users, editusers;

    public BusinessDelegate getBusinessDeletage() {
        return businessDelegate;
    }

    public void setBusinessDelegate(BusinessDelegate businessDeletage) {
        this.businessDelegate = businessDeletage;
    }

    public Object getUsers() {
        return users;
    }
}
```

```
public void setUsers(Object users) {
    this.users = users;
}

public Object getEditusers() {
    return editusers;
}

public void setEditusers(Object editusers) {
    this.editusers = editusers;
}

public User getUser() {
    return user;
}

public void setUser(User user) {
    this.user = user;
}

public void setServletRequest(HttpServletRequest request) {
    this.request = request;
}

public HttpServletRequest getServletRequest() {
    return this.request;
}

public String insertUser() throws Exception {
    Logger.debug("UserAction.insertUser()");
    this.businessDelegate.insertUser(user);
    Logger.debug("1 Record Inserted Successfully ");
    return ActionSupport.SUCCESS;
}

public String listUsers() throws Exception {
    Logger.debug("UserAction.listUsers()");
    users = this.businessDelegate.listUsers();
    return ActionSupport.SUCCESS;
}

public String deleteUser() throws Exception {
    Logger.debug("UserAction.deleteUser()");
    String userid = request.getQueryString();
    ArrayList<String> list = new ArrayList<String>();
    for (String newstr : userid.split("=")) {
        list.add(newstr);
    }
    Logger.debug("value=" + list.get(1));
    this.businessDelegate.deleteUser(Integer.parseInt(list.get(1)));
    Logger.debug("1 Record Deleted Successfully ");
    return ActionSupport.SUCCESS;
}

public String editUser() throws Exception {
    Logger.debug("UserAction.editUser()");
    String userid = request.getQueryString();
}
```

```
        ArrayList<String> list = new ArrayList<String>();
        for (String newstr : userid.split("=")) {
            list.add(newstr);
        }
        editusers = this.businessDelegate
            .editUser(Integer.parseInt(list.get(1)));
        return ActionSupport.SUCCESS;
    }

    public String updateUser() throws Exception {
        logger.debug("UserAction.updateUser()");
        this.businessDelegate.updateUser(user);
        logger.debug("1 Record Updated Successfully");
        return ActionSupport.SUCCESS;
    }
}
```

BusinessDelegate.java:

```
package com.tutorials4u.business;
import java.io.IOException;
import java.sql.SQLException;
import java.util.List;
import org.apache.log4j.Logger;
import com.tutorials4u.dao.UserDao;
import com.tutorials4u.model.User;

public class BusinessDelegate {
    private static final Logger logger = Logger
        .getLogger(BusinessDelegate.class);
    private UserDao userDao = new UserDao();

    public void insertUser(User user) throws IOException, SQLException{
        logger.debug("BusinessDelegate.insertUser()");
        new UserDao().insertUser(user);
    }

    public List<User> listUsers() throws IOException, SQLException {
        logger.debug("BusinessDelegate.listUsers()");
        List<User> users = userDao.getAllUsers();
        return users;
    }

    public void deleteUser(int userid) throws IOException, SQLException {
        logger.debug("BusinessDelegate.deleteUser()");
        userDao.deleteUser(userid);
    }

    public User editUser(int userid) throws IOException, SQLException{
        logger.debug("UserAction.editUser()");
        User user = userDao.getUserById(userid);
        return user;
    }

    public void updateUser(User user) throws IOException {
        logger.debug("UserAction.updateUser()");
        userDao.updateUser(user);
    }
}
```

```
}
```

MyBatisUtil.java:

```
package com.tutorials4u.mybatis;

import java.io.IOException;
import java.io.Reader;
import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;

public class MyBatisUtil {
    private static SqlSessionFactory factory;

    private MyBatisUtil() {
    }

    static {
        Reader reader = null;
        try {
            reader = Resources.getResourceAsReader("mybatis-config.xml");
        } catch (IOException e) {
            throw new RuntimeException(e.getMessage());
        }
        factory = new SqlSessionFactoryBuilder().build(reader);
    }

    public static SqlSessionFactory getSqlSessionFactory() {
        return factory;
    }
}
```

UserMapper.java:

```
package com.tutorials4u.mybatis.mappers;

import java.util.List;
import com.tutorials4u.model.User;

public interface UserMapper {

    public void insertUser(User user);

    public User getUserById(Integer userId);

    public List<User> getAllUsers();

    public void updateUser(User user);

    public void deleteUser(Integer userId);
}
```


UserDao.java:

```
package com.tutorials4u.dao;

import java.util.List;
import org.apache.ibatis.session.SqlSession;
import com.tutorials4u.model.User;
import com.tutorials4u.mybatis.MyBatisUtil;
import com.tutorials4u.mybatis.mappers.UserMapper;

public class UserDao {

    public void insertUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.insertUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }

    public User getUserById(Integer userId) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            return userMapper.getUserById(userId);
        } finally {
            sqlSession.close();
        }
    }

    public List<User> getAllUsers() {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            return userMapper.getAllUsers();
        } finally {
            sqlSession.close();
        }
    }

    public void updateUser(User user) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.updateUser(user);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }
}
```

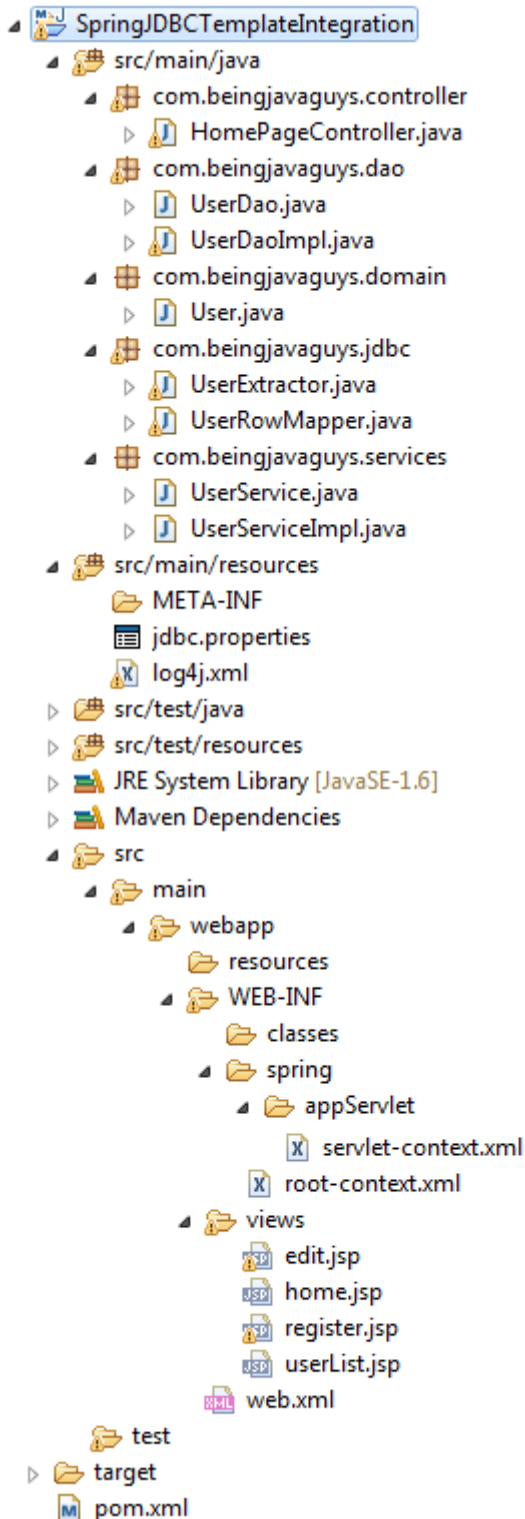
```
        }
    }

    public void deleteUser(Integer userId) {
        SqlSession sqlSession = MyBatisUtil.getSqlSessionFactory()
            .openSession();
        try {
            UserMapper userMapper =
                sqlSession.getMapper(UserMapper.class);
            userMapper.deleteUser(userId);
            sqlSession.commit();
        } finally {
            sqlSession.close();
        }
    }
}
```

Index.jsp, edituser.jsp and listUsers.jsp are same as Struts2 JDBC Integration project.

➤ Spring JDBCTemplate Integration

Project Structure:



Web.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

  <!-- The definition of the Root Spring Container shared by all Servlets and
Filters -->

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

    <!-- Creates the Spring Container shared by all Servlets and Filters -->
    <listener>
      <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
    </listener>

    <!-- Processes application requests -->
    <servlet>
      <servlet-name>appServlet</servlet-name>
      <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
      <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/spring/appServlet/servlet-
context.xml</param-value>
      </init-param>
      <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet-mapping>
      <servlet-name>appServlet</servlet-name>
      <url-pattern>/</url-pattern>
    </servlet-mapping>

</web-app>
```

Jdbc.properties:

```
app.jdbc.driverClassName=com.mysql.jdbc.Driver
app.jdbc.url=jdbc:mysql://localhost:3306/springdb
app.jdbc.username=root
app.jdbc.password=
```

Servlet-context.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc.xsd
        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">

    <!-- DispatcherServlet Context: defines this servlet's request-processing
    infrastructure -->

    <!-- Enables the Spring MVC @Controller programming model -->
    <annotation-driven />

    <context:property-placeholder location="classpath:jdbc.properties" />

    <!-- Handles HTTP GET requests for /resources/** by efficiently serving up
    static resources in the ${webappRoot}/resources directory -->
    <resources mapping="/resources/**" location="/resources/" />

    <!-- Resolves views selected for rendering by @Controllers to .jsp
    resources in the /WEB-INF/views directory -->
    <beans:bean
    class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <beans:property name="prefix" value="/WEB-INF/views/" />
        <beans:property name="suffix" value=".jsp" />
    </beans:bean>

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

    <!-- declare beans -->
    <beans:bean id="userDao" class="com.beingjavaguys.dao.UserDaoImpl" />
    <beans:bean id="userService"
    class="com.beingjavaguys.services.UserServiceImpl" />

    <!-- declare datasource bean -->
    <beans:bean id="dataSource"
    class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <beans:property name="driverClassName"
    value="${app.jdbc.driverClassName}" />
        <beans:property name="url" value="${app.jdbc.url}" />
        <beans:property name="username" value="${app.jdbc.username}" />
        <beans:property name="password" value="${app.jdbc.password}" />
    </beans:bean>

</beans:beans>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/SpringCoding/WAE/Logs/Logs.Log" />
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.beingjavaguys.domain;

public class User {

    private int userId;
    private String firstName;
    private String lastName;
    private String gender;
    private String city;

    public int getUserId() {
        return userId;
    }

    public void setUserId(int userId) {
        this.userId = userId;
    }

    public String getFirstName() {
        return firstName;
    }
}
```

```
public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getGender() {
    return gender;
}

public void setGender(String gender) {
    this.gender = gender;
}

public String getCity() {
    return city;
}

public void setCity(String city) {
    this.city = city;
}
}
```

HomeController.java:

```
package com.beingjavaguys.controller;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
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.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;

import com.beingjavaguys.domain.User;
import com.beingjavaguys.services.UserService;

@Controller
public class HomeController {

    @Autowired
    UserService userService;

    @RequestMapping("/addregister")
    public ModelAndView getRegisterForm(@ModelAttribute("user") User user,
```

```
        BindingResult result) {
            return new ModelAndView("register");
        }

@RequestMapping("/register")
public ModelAndView registerUser(@ModelAttribute User user,
                                BindingResult result) {

    List<String> genderList = new ArrayList<String>();
    genderList.add("male");
    genderList.add("female");

    List<String> cityList = new ArrayList<String>();
    cityList.add("delhi");
    cityList.add("gurgaon");
    cityList.add("meerut");
    cityList.add("noida");

    Map<String, List> map = new HashMap<String, List>();
    map.put("genderList", genderList);
    map.put("cityList", cityList);
    return new ModelAndView("register", "map", map);
}

@RequestMapping("/insert")
public String insertData(@ModelAttribute User user) {
    if (user != null)
        userService.insertData(user);
    return "redirect:/getList";
}

@RequestMapping("/getList")
public ModelAndView getUserList() {
    List<User> userList = userService.getUserList();
    return new ModelAndView("userList", "userList", userList);
}

@RequestMapping("/edit")
public ModelAndView editUser(@RequestParam String id,
                             @ModelAttribute User user) {

    user = userService.getUser(id);

    List<String> genderList = new ArrayList<String>();
    genderList.add("male");
    genderList.add("female");

    List<String> cityList = new ArrayList<String>();
    cityList.add("delhi");
    cityList.add("gurgaon");
    cityList.add("meerut");
    cityList.add("noida");

    Map<String, Object> map = new HashMap<String, Object>();
    map.put("genderList", genderList);
    map.put("cityList", cityList);
    map.put("user", user);

    return new ModelAndView("edit", "map", map);
}
```



```
    }

    @RequestMapping("/update")
    public String updateUser(@ModelAttribute User user) {
        userService.updateData(user);
        return "redirect:/getList";
    }

    @RequestMapping("/delete")
    public String deleteUser(@RequestParam String id) {
        System.out.println("id = " + id);
        userService.deleteData(id);
        return "redirect:/getList";
    }

    @RequestMapping(value = "/", method = RequestMethod.GET)
    public String home(Locale locale, Model model) {
        return "home";
    }
}
```

UserService.java:

```
package com.beingjavaguys.services;
import java.util.List;
import com.beingjavaguys.domain.User;

public interface UserService {

    public void insertData(User user);
    public List<User> getUserList();
    public void deleteData(String id);
    public User getUser(String id);
    public void updateData(User user);
}
```

UserServiceImpl.java:

```
package com.beingjavaguys.services;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import com.beingjavaguys.dao.UserDao;
import com.beingjavaguys.domain.User;

public class UserServiceImpl implements UserService {

    @Autowired
    UserDao userDao;

    @Override
    public void insertData(User user) {
        userDao.insertData(user);
    }
}
```

```
@Override
public List<User> getUserList() {
    return userDao.getUserList();
}

@Override
public void deleteData(String id) {
    userDao.deleteData(id);
}

@Override
public User getUser(String id) {
    return userDao.getUser(id);
}

@Override
public void updateData(User user) {
    userDao.updateData(user);
}
}
```

UserExtractor.java:

```
package com.beingjavaguys.jdbc;

import java.sql.ResultSet;
import java.sql.SQLException;
import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.ResultSetExtractor;
import com.beingjavaguys.domain.User;

public class UserExtractor implements ResultSetExtractor {

    public User extractData(ResultSet resultSet) throws SQLException,
        DataAccessException {

        User user = new User();

        user.setUserId(resultSet.getInt(1));
        user.setFirstName(resultSet.getString(2));
        user.setLastName(resultSet.getString(3));
        user.setGender(resultSet.getString(4));
        user.setCity(resultSet.getString(5));

        return user;
    }
}
```

UserRowMapper.java:

```
package com.beingjavaguys.jdbc;

import java.sql.ResultSet;
import java.sql.SQLException;
import org.springframework.jdbc.core.RowMapper;
import com.beingjavaguys.domain.User;

public class UserRowMapper implements RowMapper {

    public User mapRow(ResultSet resultSet, int line) throws SQLException {
        UserExtractor userExtractor = new UserExtractor();
        return userExtractor.extractData(resultSet);
    }

}
```

UserDao.java:

```
package com.beingjavaguys.dao;

import java.util.List;
import com.beingjavaguys.domain.User;

public interface UserDao {
    public void insertData(User user);
    public List<User> getUserList();
    public void updateData(User user);
    public void deleteData(String id);
    public User getUser(String id);
}
```

UserDaoImpl.java:

```
package com.beingjavaguys.dao;

import java.util.ArrayList;
import java.util.List;
import javax.sql.DataSource;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.JdbcTemplate;
import com.beingjavaguys.domain.User;
import com.beingjavaguys.jdbc.UserRowMapper;
import org.apache.log4j.Logger;

public class UserDaoImpl implements UserDao {
    @Autowired
    DataSource dataSource;
    // Get a logger
    private static final Logger logger = Logger
        .getLogger(UserDaoImpl.class);

    public void insertData(User user) {
        logger.debug("Inside UserDaoImpl.insertData()");
        String sql = "INSERT INTO user "
    }
```

```
        + "(first_name,last_name, gender, city) VALUES (?, ?,  
?,?)";  
  
        JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);  
        jdbcTemplate.update(  
            sql,  
            new Object[] { user.getFirstName(), user.getLastName(),  
                user.getGender(), user.getCity() });  
    }  
  
    public List<User> getUserList() {  
        logger.debug("Inside UserDaoImpl.getUserList()");  
        List userList = new ArrayList();  
        String sql = "select * from user";  
        JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);  
        userList = jdbcTemplate.query(sql, new UserRowMapper());  
        return userList;  
    }  
  
    @Override  
    public void deleteData(String id) {  
        logger.debug("Inside UserDaoImpl.deleteData()");  
        String sql = "delete from user where user_id=" + id;  
        JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);  
        jdbcTemplate.update(sql);  
    }  
  
    @Override  
    public void updateData(User user) {  
        logger.debug("Inside UserDaoImpl.updateData()");  
        String sql = "UPDATE user set first_name = ?,last_name = ?, gender =  
?, city = ? where user_id = ?";  
        JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);  
  
        jdbcTemplate.update(  
            sql,  
            new Object[] { user.getFirstName(), user.getLastName(),  
                user.getGender(), user.getCity(),  
user.getUserId() });  
    }  
  
    @Override  
    public User getUser(String id) {  
        logger.debug("Inside UserDaoImpl.getUser()");  
        List<User> userList = new ArrayList<User>();  
        String sql = "select * from user where user_id= " + id;  
        JdbcTemplate jdbcTemplate = new JdbcTemplate(dataSource);  
        userList = jdbcTemplate.query(sql, new UserRowMapper());  
        return userList.get(0);  
    }  
}
```

Home.jsp:

```
<%response.sendRedirect("getList");%>
```

Register.jsp:

```
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<title>Spring JDBCTemplate Integration</title>
<style>
body {
    font-size: 20px;
    color: teal;
    font-family: Calibri;
}

td {
    font-size: 15px;
    color: black;
    width: 100px;
    height: 22px;
    text-align: left;
}

.heading {
    font-size: 18px;
    color: white;
    font: bold;
    background-color: orange;
    border: thick;
}
</style>
</head>
<body>
    <center>
        <b>Registration Form </b>
        <div>
            <form:form method="post"
            action="/SpringJDBCTemplateIntegration/insert"
            modelAttribute="user">
                <table>
                    <tr>
                        <td>First Name :</td>
                        <td><form:input path="firstName" /></td>
                    </tr>
                    <tr>
                        <td>Last Name :</td>
                        <td><form:input path="lastName" /></td>
                    </tr>
                    <tr>
                        <td>Gender :</td>
                        <td><form:radiobuttons path="gender"
                            items="${map.genderList}"
                        /></td>
                    </tr>
                </table>
            </form>
        </div>
    </center>
</body>
```

```

                                <tr>
                                    <td>City :</td>
                                    <td><form:select path="city"
items="${map.cityList}" /></td>
                                </tr>
                                <tr>
                                    <td> </td>
                                    <td><input type="submit" value="Save"
/></td>
                                </tr>
                                <tr>
                                    <td colspan="2"><a href="getList">Click Here to
See User List</a></td>
                                </tr>
                            </table>
                        </form:form>
                    </div>
                </center>
            </body>
        </html>

```

userList.jsp:

```

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<title>Spring JDBCTemplate Integration</title>
<style>
body {
    font-size: 20px;
    color: teal;
    font-family: Calibri;
}

td {
    font-size: 15px;
    color: black;
    width: 100px;
    height: 22px;
    text-align: center;
}

.heading {
    font-size: 18px;
    color: white;
    font: bold;
    background-color: orange;
    border: thick;
}
</style>
</head>
<body>
    <center>
        <b>User List </b>
    </center>

```

```
<table border="1">
  <tr>
    <td class="heading">User Id</td>
    <td class="heading">First Name</td>
    <td class="heading">Last Name</td>
    <td class="heading">Gender</td>
    <td class="heading">City</td>
    <td class="heading">Edit</td>
    <td class="heading">Delete</td>
  </tr>
  <c:forEach var="user" items="${userList}">
    <tr>
      <td>${user.userId}</td>
      <td>${user.firstName}</td>
      <td>${user.lastName}</td>
      <td>${user.gender}</td>
      <td>${user.city}</td>
      <td><a href="edit?id=${user.userId}">Edit</a></td>
      <td><a href="delete?id=${user.userId}">Delete</a></td>
    </tr>
  </c:forEach>
  <tr><td colspan="7"><a href="/SpringJDBCTemplateIntegration/register">Add New
User</a></td></tr>
</table>
</center>
</body>
</html>
```

Edit.jsp:

```
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://www.springframework.org/tags" prefix="spring"%>
<html>
<head>
<title>Spring JDBCTemplate Integration</title>
<style>
body {
    font-size: 20px;
    color: teal;
    font-family: Calibri;
}

td {
    font-size: 15px;
    color: black;
    width: 100px;
    height: 22px;
    text-align: center;
}

.heading {
    font-size: 18px;
    color: white;
    font: bold;
    background-color: orange;
    border: thick;
}
```

```
</style>
</head>
<body>
    <b>Edit User Details </b>
    <div>
        <form:form method="post"
            action="/SpringJDBCTemplateIntegration/update"
modelAttribute="user">
            <table>
                <tr>
                    <td>First Name :</td>
                    <td><form:input path="firstName"
                        value="${map.user.firstName}"
                    /></td>
                </tr>
                <tr>
                    <td>Last Name :</td>
                    <td><form:input path="lastName"
value="${map.user.lastName}" />
                    </td>
                </tr>
                <tr>
                    <td>Gender :</td>
                    <td><spring:bind path="gender">
                        <c:forEach
items='${map.genderList}' var='genderName'>
                            <c:choose>
                                <c:when
test="${genderName eq map.user.gender}">
                                    <input
type="radio" name="gender" value="${genderName}"
checked="checked">${genderName}
                                </c:when>
                                <c:otherwise>
                                    <input
type="radio" name="gender" value="${genderName}">${genderName}
                                </c:otherwise>
                            </c:choose>
                        </c:forEach>
                    </spring:bind></td>
                </tr>
                <tr>
                    <td>City :</td>
                    <td><spring:bind path="city">
                        <select name="city">
                            <c:forEach
items='${map.cityList}' var='cityName'>
                                <c:choose>
                                    <c:when
test="${cityName eq map.user.city}">
                                        <option value="${cityName}" selected="true">${cityName}</option>
                                    </c:when>
                                    <c:otherwise>
                                        <option value="${cityName}">${cityName}</option>
                                    </c:otherwise>
                                </c:choose>
                            </c:forEach>
                        </select>
                    </spring:bind></td>
                </tr>
            </table>
        </form:form>
    </div>
</body>
```



```
</c:otherwise>

                                                                </c:choose>
                                                                </c:forEach>
                                                                </select>
                                                                </spring:bind></td>

</tr>
<tr>
    <td> </td>
    <td><input type="submit" value="Save"

</tr>
</table>
<form:hidden path="userId" value="${map.user.userId}"

/>

</form:form>
</div>

</body>
</html>
```

Registration Form

First Name :

Last Name :

Gender : ☒ male ☐ female

City :

[Click Here to See User List](#)

User Id	First Name	Last Name	Gender	City	Edit	Delete
1	Amit	Bhalerao	male	delhi	Edit	Delete
2	Kasturi Amit	Bhalerao	female	noida	Edit	Delete
3	Amey	Bhalerao	male	meerut	Edit	Delete
6	Avani	Bhalerao	female	gurgaon	Edit	Delete
9	Test	Test	male	gurgaon	Edit	Delete
Add New User						

Edit User Details

First Name :

Last Name :

Gender : ☒ male ☐ female

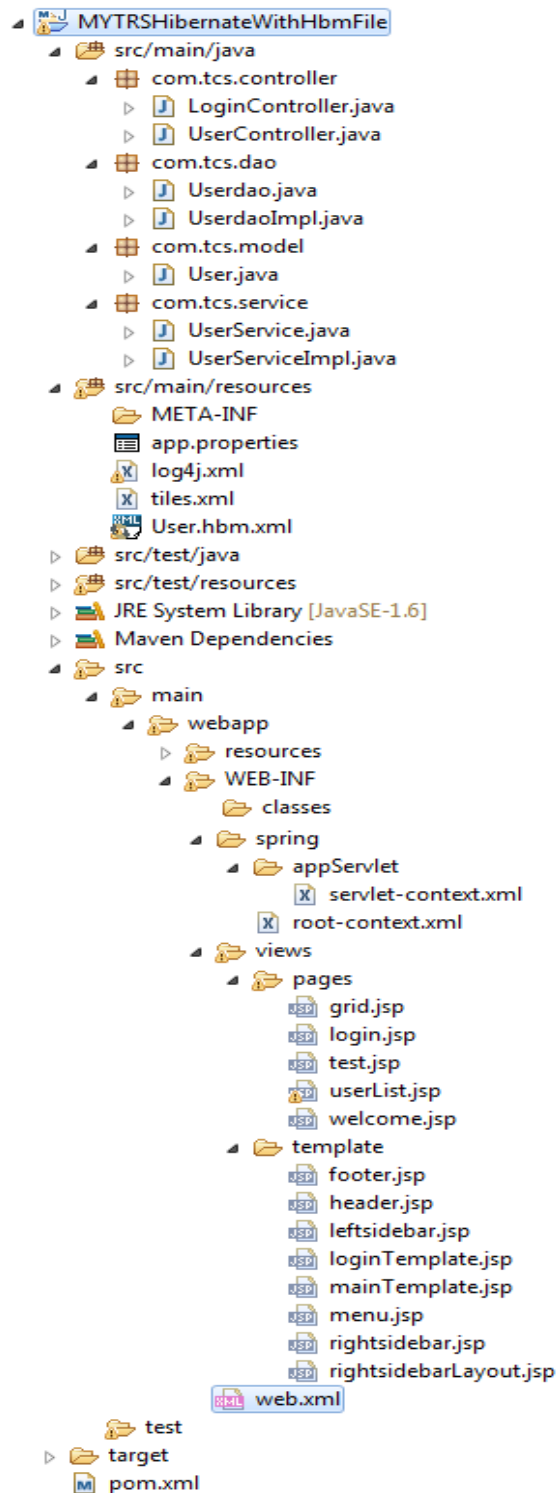
City :

User Id	First Name	Last Name	Gender	City	Edit	Delete
1	Amit	Bhalerao	male	delhi	Edit	Delete
2	Kasturi Amit	Bhalerao	female	noida	Edit	Delete
3	Amey	Bhalerao	male	meerut	Edit	Delete
6	Avani	Bhalerao	female	gurgaon	Edit	Delete
9	Test11111	Test111	male	meerut	Edit	Delete
Add New User						

➤ Spring Hibernate Integration

1) Using XML Mapping Configuration

Project Structure:



Pom.xml:

```
<dependencies>
  <!-- Spring -->
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${org.springframework-version}</version>
    <exclusions>
      <!-- Exclude Commons Logging in favor of SLF4j -->
      <exclusion>
        <groupId>commons-logging</groupId>
        <artifactId>commons-logging</artifactId>
      </exclusion>
    </exclusions>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-orm</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-web</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <!-- AspectJ -->
  <dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
  </dependency>
```

```
<!-- Logging -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>${org.slf4j-version}</version>
</dependency>

<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>jcl-over-slf4j</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>

<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>

<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.15</version>
  <exclusions>
    <exclusion>
      <groupId>javax.mail</groupId>
      <artifactId>mail</artifactId>
    </exclusion>
    <exclusion>
      <groupId>javax.jms</groupId>
      <artifactId>jms</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jdmk</groupId>
      <artifactId>jmxtools</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jmx</groupId>
      <artifactId>jmxri</artifactId>
    </exclusion>
  </exclusions>
  <scope>runtime</scope>
</dependency>

<!-- @Inject -->
<dependency>
  <groupId>javax.inject</groupId>
  <artifactId>javax.inject</artifactId>
  <version>1</version>
</dependency>

<!-- Servlet -->
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
  <scope>provided</scope>
</dependency>
```

```
<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <version>2.1</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>

<!-- Test -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.7</version>
  <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.9</version>
</dependency>

<!-- Hibernate framework -->
<dependency>
  <groupId>org.hibernate</groupId>
  <artifactId>hibernate-core</artifactId>
  <version>4.2.4.Final</version>
</dependency>

<!-- Tiles dependencies -->
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-api</artifactId>
  <version>2.2.2</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-autotag-core-runtime</artifactId>
  <version>1.1.0</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-core</artifactId>
  <version>2.2.2</version>
</dependency>
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-jsp</artifactId>
  <version>2.2.2</version>
</dependency>
```

```
<dependency>
    <groupId>org.apache.tiles</groupId>
    <artifactId>tiles-servlet</artifactId>
    <version>2.2.2</version>
</dependency>

<dependency>
    <groupId>org.apache.tiles</groupId>
    <artifactId>tiles-template</artifactId>
    <version>2.2.2</version>
</dependency>

</dependencies>
```

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"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/spring/root-context.xml</param-value>
    </context-param>
    <listener>
        <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-
class>
        </listener>
    <servlet>
        <servlet-name>appServlet</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-
value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>appServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>
```

Servlet-context.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:p="http://www.springframework.org/schema/p"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc.xsd
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd">

    <annotation-driven />

    <resources mapping="/resources/**" location="/resources/mytheme/" />
    <context:property-placeholder location="classpath:app.properties" />
    <context:component-scan base-package="com.tcs" />

    <beans:bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <beans:property name="driverClassName" value="${app.jdbc.driverClassName}"
/>

        <beans:property name="url" value="${app.jdbc.url}" />
        <beans:property name="username" value="${app.jdbc.username}" />
        <beans:property name="password" value="${app.jdbc.password}" />
    </beans:bean>

    <beans:bean id="sessionFactory"
class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">
        <beans:property name="dataSource">
            <beans:ref bean="dataSource" />
        </beans:property>
        <beans:property name="packagesToScan" value="com.tcs.model">
        </beans:property>
        <beans:property name="hibernateProperties">
            <beans:props>
                <beans:prop key="hibernate.hbm2ddl.auto">update
                </beans:prop>
                <beans:prop key="hibernate.show_sql">true
                </beans:prop>
                <beans:prop key="hibernate.dialect">
org.hibernate.dialect.MySQLDialect
                </beans:prop>
            </beans:props>
        </beans:property>
        <beans:property name="mappingResources">
            <beans:list>
                <beans:value>User.hbm.xml</beans:value>
            </beans:list>
        </beans:property>
    </beans:bean>
```

```
<beans:bean id="transactionManager"
class="org.springframework.orm.hibernate4.HibernateTransactionManager">
    <beans:property name="sessionFactory">
        <beans:ref bean="sessionFactory" />
    </beans:property>
</beans:bean>

<tx:annotation-driven transaction-manager="transactionManager" />

<!-- Tiles Configuration -->

<beans:bean id="tilesConfigurer"
class="org.springframework.web.servlet.view.tiles2.TilesConfigurer"
p:definitions="classpath:tiles.xml" />

<beans:bean id="tilesViewResolver"
class="org.springframework.web.servlet.view.UrlBasedViewResolver"
p:viewClass="org.springframework.web.servlet.view.tiles2.TilesView"
/>

</beans:beans>
```

App.properties:

```
app.jdbc.driverClassName=com.mysql.jdbc.Driver
app.jdbc.url=jdbc:mysql://localhost:3306/mybatis
app.jdbc.username=root
app.jdbc.password=
```

tiles.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE tiles-definitions PUBLIC
    "-//Apache Software Foundation//DTD Tiles Configuration 2.0//EN"
    "http://tiles.apache.org/dtds/tiles-config_2_0.dtd">

<tiles-definitions>

    <definition name="LoginTemplate" template="/WEB-INF/views/template/LoginTemplate.jsp">
        <put-attribute name="body" value=""></put-attribute>
    </definition>

    <definition extends="LoginTemplate" name="Login">
        <put-attribute name="title" value="Login :: MyTRS"></put-attribute>
        <put-attribute name="body" value="/WEB-INF/views/pages/Login.jsp"></put-attribute>
    </definition>

    <definition name="base.definition" template="/WEB-INF/views/template/mainTemplate.jsp">
        <put-attribute name="title" value=""></put-attribute>
        <put-attribute name="header" value="/WEB-INF/views/template/header.jsp"></put-attribute>
        <put-attribute name="menu" value="/WEB-INF/views/template/menu.jsp"></put-attribute>
        <put-attribute name="body" value=""></put-attribute>
    </definition>

</tiles-definitions>
```



```
<put-attribute name="sidebar" value="/WEB-INF/views/template/leftsidebar.jsp"></put-attribute>
<put-attribute name="footer" value="/WEB-INF/views/template/footer.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="welcome">
  <put-attribute name="title" value="Dashboard"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/welcome.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="userList">
  <put-attribute name="title" value="List Of Users"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/userList.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="edit">
  <put-attribute name="title" value="Edit User Details"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/edit.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="grid">
  <put-attribute name="title" value="List Of Users"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/grid.jsp"></put-attribute>
</definition>

<definition name="rightsidebarLayout" template="/WEB-INF/views/template/rightsidebarLayout.jsp">
  <put-attribute name="title" value=""></put-attribute>
  <put-attribute name="header" value="/WEB-INF/views/template/header.jsp"></put-attribute>
  <put-attribute name="menu" value="/WEB-INF/views/template/menu.jsp"></put-attribute>
  <put-attribute name="body" value=""></put-attribute>
  <put-attribute name="rightsidebar" value="/WEB-INF/views/template/rightsidebar.jsp"></put-attribute>
  <put-attribute name="footer" value="/WEB-INF/views/template/footer.jsp"></put-attribute>
</definition>

<definition extends="rightsidebarLayout" name="test">
  <put-attribute name="title" value="List Of Users"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/test.jsp"></put-attribute>
</definition>

</tiles-definitions>
```

User.hbm.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD//EN"
    "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.tcs.model.User" table="MYUSER">
        <meta attribute="class-description">
            This class contains the User detail.
        </meta>
        <id name="user_id" type="int" column="USER_ID">
            <generator class="native"/>
        </id>
        <property name="username" column="USERNAME" type="string"/>
        <property name="password" column="PASSWORD" type="string"/>
        <property name="firstName" column="FIRSTNAME" type="string"/>
        <property name="lastName" column="LASTNAME" type="string"/>
        <property name="phone" column="PHONE" type="string"/>
        <property name="emailId" column="EMAILID" type="string"/>
    </class>
</hibernate-mapping>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/SpringCoding/WAE/Logs/Logs.log" />
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```

User.java:

```
package com.tcs.model;
import javax.persistence.Transient;

public class User {
    private int user_id;
    private String username;
    private String password;
    private String firstName;
    private String lastName;
    private String phone;
    private String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }
}
```

```
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }

    @Override
    public String toString() {
        return "Id: " + user_id + "; fname : " + firstName + " phone : " + phone
+ "; emailId: " + emailId;
    }
}
```

UserController.java:

```
package com.tcs.controller;

import java.util.List;
import java.util.Locale;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import com.tcs.model.User;
import com.tcs.service.UserService;

@Controller
public class UserController {

    @Autowired
    UserService service;
```

```
@RequestMapping(value = "/save", method = RequestMethod.POST)
public String save(@ModelAttribute("user") User user,
                  BindingResult br, Model model) {
    if (br.hasErrors()) {
        return "welcome";
    }
    if (user.getUser_id() > 0) {
        service.updateUser(user);
        model.addAttribute("showUpdateMessage", "true");
    } else {
        service.addUser(user);
        model.addAttribute("showSaveMessage", "true");
    }

    return "redirect:userList";
}

@RequestMapping(value = { "userList" }, method = RequestMethod.GET)
public String list(Model model) {
    List<User> userlist = service.listUser();
    model.addAttribute("userlist", userlist);
    return "userList";
}

@RequestMapping(value = "/edit/{id}", method = RequestMethod.GET)
public String edit(@PathVariable Integer id, Model model) {
    model.addAttribute("showUpdateMessage", "true");
    model.addAttribute("user", service.getUser(id));
    return "welcome";
}

@RequestMapping(value = "/delete/{id}", method = RequestMethod.GET)
public String delete(@PathVariable Integer id, Model model) {
    service.deleteUser(id);
    model.addAttribute("showDeleteMessage", "true");
    return "redirect:/userList";
}
}
```

UserService.java:

```
package com.tcs.service;

import java.util.List;
import com.tcs.model.User;

public interface UserService {
    public User getUser(int id);
    public void updateUser(User person);
    public List<User> listUser();
    public void deleteUser(int id);
    public void addUser(User person);
    public User validateUser(User usr);
}
```

UserServiceImpl.java:

```
package com.tcs.service;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import com.tcs.dao.Userdao;
import com.tcs.model.User;

@Service
@Transactional
public class UserServiceImpl implements UserService {

    @Autowired
    Userdao userDao;

    @Override
    public User getUser(int id) {
        return userDao.getUser(id);
    }

    @Override
    public void updateUser(User user) {
        userDao.updateUser(user);
    }

    @Override
    public List<User> listUser() {
        return userDao.listUsers();
    }

    @Override
    public void deleteUser(int id) {
        userDao.deleteUser(id);
    }

    @Override
    public void addUser(User user) {
        userDao.addUser(user);
    }

    @Override
    public User validateUser(User usr) {
        return userDao.validateUser(usr);
    }
}
```

UserDao.java:

```
package com.tcs.dao;
import java.util.List;
import com.tcs.model.User;

public interface UserDao {
    public User getUser(int id);
    public List<User> listUsers();
    public void deleteUser(int id);
    public void addUser(User user);
    public void updateUser(User user);
    public User validateUser(User user);
}
```

UserDaoImpl.java:

```
package com.tcs.dao;

import java.util.List;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import com.tcs.model.User;

@Repository
public class UserDaoImpl implements UserDao {

    @Autowired
    private SessionFactory sessionFactory;

    public Session getCurrentSession() {
        return sessionFactory.getCurrentSession();
    }

    @Override
    public User getUser(int id) {
        User user = (User) getCurrentSession().get(User.class, id);
        return user;
    }

    @SuppressWarnings("unchecked")
    @Override
    public List<User> listUsers() {
        Query query = getCurrentSession().createQuery("from User");
        List<User> list = query.list();
        return list;
    }

    @Override
    public void deleteUser(int id) {
        User user = getUser(id);
        getCurrentSession().delete(user);
    }
}
```

```
@Override
public void addUser(User user) {
    getCurrentSession().save(user);
}

@Override
public void updateUser(User user) {
    getCurrentSession().update(user);
}

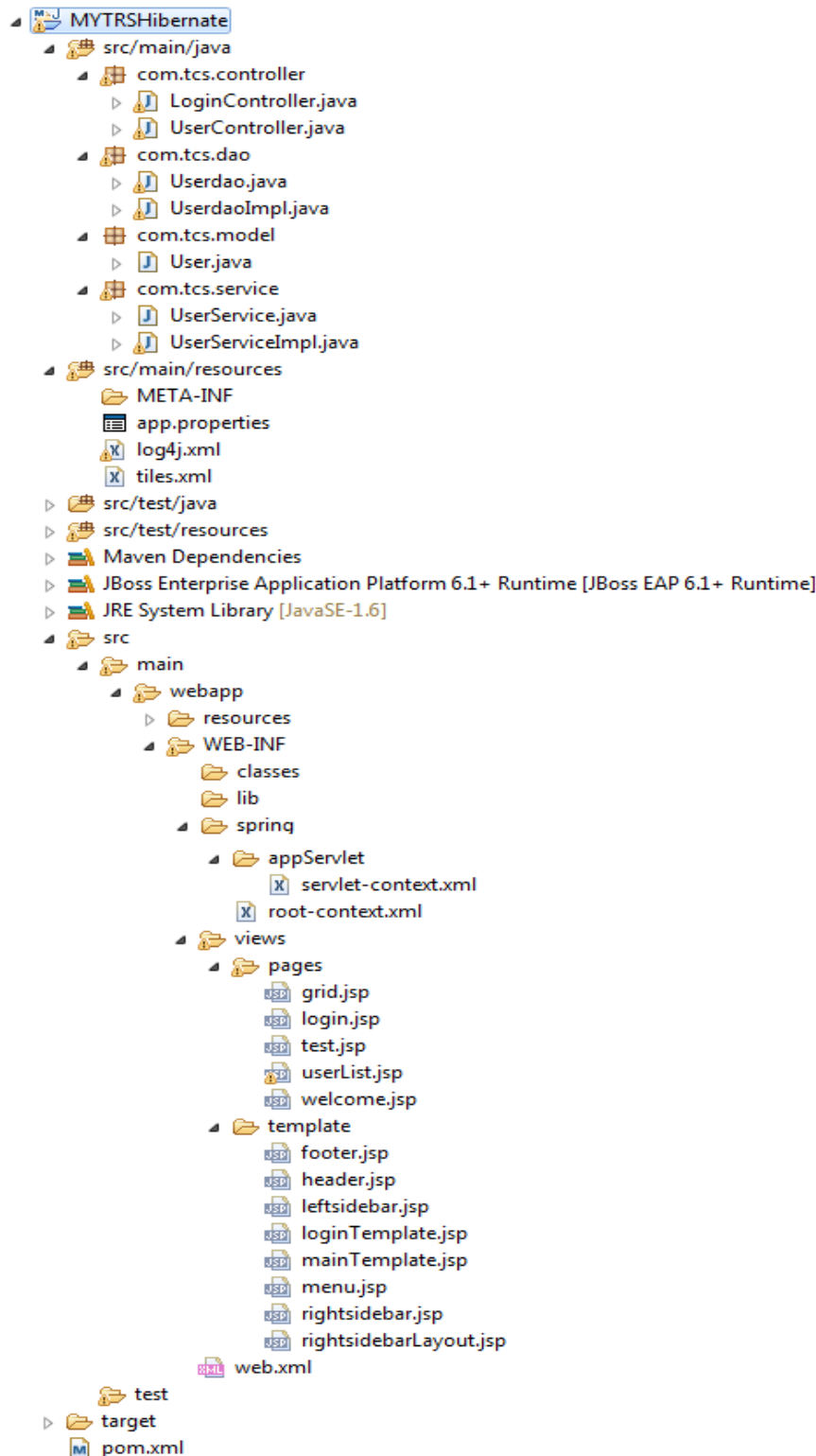
@SuppressWarnings("rawtypes")
@Override
public User validateUser(User usr) {
    Query query = getCurrentSession().createQuery("from User where
    username= :username and password= :password");

    query.setParameter("username",usr.getUsername());
    query.setParameter("password",usr.getPassword());

    List list = query.list();
    if (list.size() > 0) {
        usr = (User) list.get(0);
        usr.setValid(true);
    } else {
        usr.setValid(false);
    }
    return usr;
}
}
```


2) Using Annotations

Project Structure:



Pom.xml:

```
<dependencies>
  <!-- Spring -->
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${org.springframework-version}</version>
    <exclusions>
      <!-- Exclude Commons Logging in favor of SLF4j -->
      <exclusion>
        <groupId>commons-logging</groupId>
        <artifactId>commons-logging</artifactId>
      </exclusion>
    </exclusions>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-orm</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-web</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <!-- AspectJ -->
  <dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
  </dependency>
```

```
<!-- Logging -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>${org.slf4j-version}</version>
</dependency>

<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>jcl-over-slf4j</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>

<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>

<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.15</version>
  <exclusions>
    <exclusion>
      <groupId>javax.mail</groupId>
      <artifactId>mail</artifactId>
    </exclusion>
    <exclusion>
      <groupId>javax.jms</groupId>
      <artifactId>jms</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jdmk</groupId>
      <artifactId>jmxtools</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jmx</groupId>
      <artifactId>jmxri</artifactId>
    </exclusion>
  </exclusions>
  <scope>runtime</scope>
</dependency>

<!-- @Inject -->
<dependency>
  <groupId>javax.inject</groupId>
  <artifactId>javax.inject</artifactId>
  <version>1</version>
</dependency>

<!-- Servlet -->
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
  <scope>provided</scope>
</dependency>
```

```
<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <version>2.1</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>

<!-- Test -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.7</version>
  <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.9</version>
</dependency>

<!-- Hibernate framework -->
<dependency>
  <groupId>org.hibernate</groupId>
  <artifactId>hibernate-core</artifactId>
  <version>4.2.4.Final</version>
</dependency>

<!-- Tiles dependencies -->
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-api</artifactId>
  <version>2.2.2</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-autotag-core-runtime</artifactId>
  <version>1.1.0</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-core</artifactId>
  <version>2.2.2</version>
</dependency>
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-jsp</artifactId>
  <version>2.2.2</version>
</dependency>
```

```
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-servlet</artifactId>
  <version>2.2.2</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-template</artifactId>
  <version>2.2.2</version>
</dependency>
```

```
</dependencies>
```

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"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/root-context.xml</param-value>
  </context-param>
  <listener>
    <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
    </listener>
  <servlet>
    <servlet-name>appServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-
class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>appServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>
</web-app>
```

Servlet-context.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:p="http://www.springframework.org/schema/p"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc.xsd
    http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
    http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx.xsd
    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd">

    <annotation-driven />

    <resources mapping="/resources/**" location="/resources/mytheme/" />
    <context:property-placeholder location="classpath:app.properties" />
    <context:component-scan base-package="com.tcs" />

    <beans:bean id="dataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <beans:property name="driverClassName" value="${app.jdbc.driverClassName}"
/>

        <beans:property name="url" value="${app.jdbc.url}" />
        <beans:property name="username" value="${app.jdbc.username}" />
        <beans:property name="password" value="${app.jdbc.password}" />
    </beans:bean>

    <beans:bean id="sessionFactory"
        class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">
        <beans:property name="dataSource">
            <beans:ref bean="dataSource" />
        </beans:property>
        <beans:property name="packagesToScan" value="com.tcs.model">
        </beans:property>
        <beans:property name="hibernateProperties">
            <beans:props>
                <beans:prop key="hibernate.hbm2ddl.auto">
                    update
                </beans:prop>
                <beans:prop key="hibernate.show_sql">
                    true
                </beans:prop>
                <beans:prop key="hibernate.dialect">
                    org.hibernate.dialect.MySQLDialect
                </beans:prop>
            </beans:props>
        </beans:property>
    </beans:bean>
```

```
<beans:bean id="transactionManager"
class="org.springframework.orm.hibernate4.HibernateTransactionManager">
    <beans:property name="sessionFactory">
        <beans:ref bean="sessionFactory" />
    </beans:property>
</beans:bean>

<tx:annotation-driven transaction-manager="transactionManager" />

<!-- Tiles Configuration -->

<beans:bean id="tilesConfigurer"
class="org.springframework.web.servlet.view.tiles2.TilesConfigurer"
p:definitions="classpath:tiles.xml" />

<beans:bean id="tilesViewResolver"
class="org.springframework.web.servlet.view.UrlBasedViewResolver"
p:viewClass="org.springframework.web.servlet.view.tiles2.TilesView"
/>

</beans:beans>
```

App.properties:

```
app.jdbc.driverClassName=com.mysql.jdbc.Driver
app.jdbc.url=jdbc:mysql://localhost:3306/mybatis
app.jdbc.username=root
app.jdbc.password=
```

tiles.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE tiles-definitions PUBLIC
    "-//Apache Software Foundation//DTD Tiles Configuration 2.0//EN"
    "http://tiles.apache.org/dtds/tiles-config_2_0.dtd">

<tiles-definitions>
    <definition name="LoginTemplate" template="/WEB-INF/views/template/LoginTemplate.jsp">
        <put-attribute name="body" value=""></put-attribute>
    </definition>

    <definition extends="LoginTemplate" name="Login">
        <put-attribute name="title" value="Login :: MyTRS"></put-attribute>
        <put-attribute name="body" value="/WEB-INF/views/pages/Login.jsp">
            </put-attribute>
        </definition>

        <definition name="base.definition" template="/WEB-INF/views/template/mainTemplate.jsp">
            <put-attribute name="title" value=""></put-attribute>
            <put-attribute name="header" value="/WEB-INF/views/template/header.jsp"></put-attribute>
            <put-attribute name="menu" value="/WEB-INF/views/template/menu.jsp">
                </put-attribute>
            <put-attribute name="body" value=""></put-attribute>
```

```
<put-attribute name="sidebar" value="/WEB-INF/views/template/leftsidebar.jsp"></put-attribute>
<put-attribute name="footer" value="/WEB-INF/views/template/footer.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="welcome">
  <put-attribute name="title" value="Dashboard"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/welcome.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="userList">
  <put-attribute name="title" value="List Of Users"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/userList.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="edit">
  <put-attribute name="title" value="Edit User Details"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/edit.jsp"></put-attribute>
</definition>

<definition extends="base.definition" name="grid">
  <put-attribute name="title" value="List Of Users"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/grid.jsp"></put-attribute>
</definition>

<definition name="rightsidebarLayout" template="/WEB-INF/views/template/rightsidebarLayout.jsp">
  <put-attribute name="title" value=""></put-attribute>
  <put-attribute name="header" value="/WEB-INF/views/template/header.jsp"></put-attribute>
  <put-attribute name="menu" value="/WEB-INF/views/template/menu.jsp"></put-attribute>
  <put-attribute name="body" value=""></put-attribute>
  <put-attribute name="rightsidebar" value="/WEB-INF/views/template/rightsidebar.jsp"></put-attribute>
  <put-attribute name="footer" value="/WEB-INF/views/template/footer.jsp"></put-attribute>
</definition>

<definition extends="rightsidebarLayout" name="test">
  <put-attribute name="title" value="List Of Users"></put-attribute>
  <put-attribute name="body" value="/WEB-INF/views/pages/test.jsp"></put-attribute>
</definition>

</tiles-definitions>
```


Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
  xmlns:log4j='http://jakarta.apache.org/Log4j/'>

  <!-- Console -->
  <appender name="console" class="org.apache.Log4j.ConsoleAppender">
    <layout class="org.apache.Log4j.PatternLayout">
      <param name="ConversionPattern"
        value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
  </appender>

  <!-- file -->
  <appender name="file" class="org.apache.Log4j.RollingFileAppender">
    <param name="append" value="false" />
    <param name="maxFileSize" value="10KB" />
    <param name="maxBackupIndex" value="5" />
    <param name="file" value="F:/Coding/SpringCoding/WAE/Logs/Logs.Log" />
    <layout class="org.apache.Log4j.PatternLayout">
      <param name="ConversionPattern"
        value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
    </layout>
  </appender>

  <root>
    <level value="DEBUG" />
    <appender-ref ref="console" />
    <appender-ref ref="file" />
  </root>
</log4j:configuration>
```

User.java:

```
package com.tcs.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.Transient;

@Entity
@Table(name = "myuser")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(unique = true, nullable = false)
    private int user_id;
    @Column
    String username;
    @Column
    String password;
    @Column
    String firstName;
    @Column
    String lastName;
    @Column
    String phone;
    @Column
    String emailId;
    private boolean valid;

    @Transient
    public boolean isValid() {
        return valid;
    }

    public void setValid(boolean valid) {
        this.valid = valid;
    }

    public int getUser_id() {
        return user_id;
    }

    public void setUser_id(int user_id) {
        this.user_id = user_id;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }
}
```

```
    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public String getEmailId() {
        return emailId;
    }

    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }

    @Override
    public String toString() {
        return "Id: " + user_id + "; fname : " + firstName + "' phone : "
            + phone + "; emailId: " + emailId;
    }
}
```

UserController.java:

```
package com.tcs.controller;

import java.util.List;
import java.util.Locale;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import com.tcs.model.User;
import com.tcs.service.UserService;

@Controller
public class UserController {
    @Autowired
    UserService service;

    @RequestMapping(value = "/save", method = RequestMethod.POST)
    public String save(@ModelAttribute("user") User user,
        BindingResult br, Model model) {
        if (br.hasErrors()) {
            return "welcome";
        }
        if (user.getUser_id() > 0) {
            service.updateUser(user);
            model.addAttribute("showUpdateMessage", "true");
        } else {
            service.addUser(user);
            model.addAttribute("showSaveMessage", "true");
        }

        return "redirect:userList";
    }

    @RequestMapping(value = { "userList" }, method = RequestMethod.GET)
    public String list(Model model) {
        List<User> userlist = service.listUser();
        model.addAttribute("userlist", userlist);
        return "userList";
    }

    @RequestMapping(value = "/edit/{id}", method = RequestMethod.GET)
    public String edit(@PathVariable Integer id, Model model) {
        model.addAttribute("showUpdateMessage", "true");
        model.addAttribute("user", service.getUser(id));
        return "welcome";
    }

    @RequestMapping(value = "/delete/{id}", method = RequestMethod.GET)
    public String delete(@PathVariable Integer id, Model model) {
        service.deleteUser(id);
        model.addAttribute("showDeleteMessage", "true");
        return "redirect:/userList";
    }
}
```

```
@RequestMapping(value = "grid", method = RequestMethod.GET)
public String showGrid(Locale locale, Model model) {
    List<User> users = service.listUser();
    model.addAttribute("users", users);
    return "grid";
}
}
```

UserService.java:

```
package com.tcs.service;

import java.util.List;
import com.tcs.model.User;

public interface UserService {
    public User getUser(int id);
    public void updateUser(User user);
    public List<User> listUser();
    public void deleteUser(int id);
    public void addUser(User user);
    public User validateUser(User usr);
}
```

UserServiceImpl.java:

```
package com.tcs.service;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import com.tcs.dao.Userdao;
import com.tcs.model.User;

@Service
@Transactional
public class UserServiceImpl implements UserService {

    @Autowired
    Userdao userDao;

    @Override
    public User getUser(int id) {
        return userDao.getUser(id);
    }

    @Override
    public void updateUser(User user) {
        userDao.updateUser(user);
    }

    @Override
    public List<User> listUser() {
        return userDao.listUsers();
    }
}
```

```
@Override
public void deleteUser(int id) {
    userDao.deleteUser(id);
}

@Override
public void addUser(User user) {
    userDao.addUser(user);
}

@Override
public User validateUser(User usr) {
    return userDao.validateUser(usr);
}
}
```

UserDao.java:

```
package com.tcs.dao;

import java.util.List;
import com.tcs.model.User;

public interface UserDao {
    public User getUser(int id);
    public List<User> listUsers();
    public void deleteUser(int id);
    public void addUser(User user);
    public void updateUser(User user);
    public User validateUser(User user);
}
```

UserDaoImpl.java:

```
package com.tcs.dao;

import java.util.List;
import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import com.tcs.model.User;

@Repository
public class UserDaoImpl implements UserDao {

    @Autowired
    private SessionFactory sessionFactory;

    public Session getCurrentSession() {
        return sessionFactory.getCurrentSession();
    }
}
```

```
@Override
public User getUser(int id) {
    User user = (User) getCurrentSession().get(User.class, id);
    return user;
}

@Override
public List<User> listUsers() {
    return getCurrentSession().createQuery("From User").list();
}

@Override
public void deleteUser(int id) {
    User user = getUser(id);
    getCurrentSession().delete(user);
}


@Override
public void addUser(User user) {
    getCurrentSession().save(user);
}

@Override
public void updateUser(User user) {
    getCurrentSession().update(user);
}

@Override
public User validateUser(User usr) {
    Query query = getCurrentSession().createQuery("from User where
    username= :username and password= :password");

    query.setParameter("username",usr.getUsername());
    query.setParameter("password",usr.getPassword());

    List list = query.list();
    if (list.size() > 0) {
        usr = (User) list.get(0);
        usr.setValid(true);
    } else {
        usr.setValid(false);
    }
    return usr;
}
}
```

 Add New User

*First Name

Mrs.Kasturi Amit

*Last Name

Bhalerao

*Username

kasturi

*Password


*Phone

7756090718

*Email Id









kasturi.bhalerao@gmail.com

Save

 List Of Users

Show 10 entries

Search:

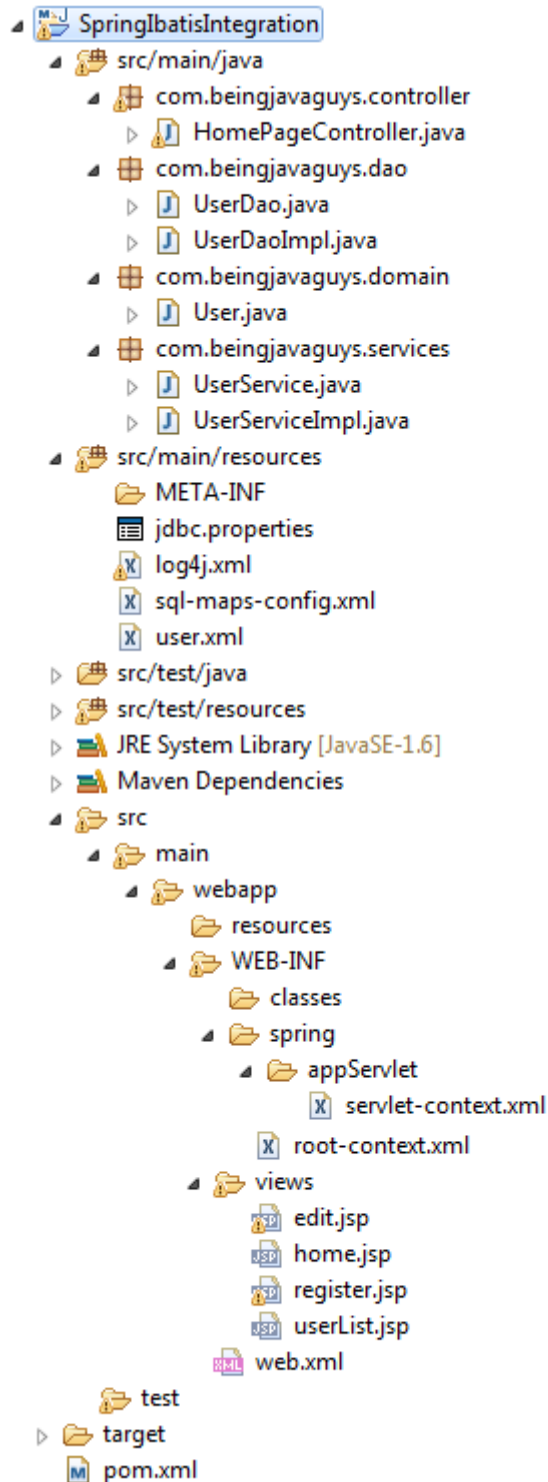
Username	First Name	Last Name	Phone	Email Id	Actions
abhijit	Abhijit	Shinde	9860960529	abhijit.shinde22@gmail.com	 
amey	Amey Amit	Bhalerao	8796154725	amey.bhalerao@wipro.com	 
amit	Amit	Bhalerao	8796154725	amit.bhalerao87@gmail.com	 
kasturi	Mrs.Kasturi Amit	Bhalerao	7756090718	kasturi.bhalerao@gmail.com	 

Showing 1 to 4 of 4 entries

FirstPrevious1NextLast

➤ Spring iBatis Integration

Project Structure:



Pom.xml:

```
<dependencies>
  <!-- Spring -->
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${org.springframework-version}</version>
    <exclusions>
      <!-- Exclude Commons Logging in favor of SLF4j -->
      <exclusion>
        <groupId>commons-logging</groupId>
        <artifactId>commons-logging</artifactId>
      </exclusion>
    </exclusions>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-orm</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-web</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${org.springframework-version}</version>
  </dependency>

  <!-- AspectJ -->
  <dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
  </dependency>
```

```
<!-- Logging -->
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>${org.slf4j-version}</version>
</dependency>

<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>jcl-over-slf4j</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>

<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <version>${org.slf4j-version}</version>
  <scope>runtime</scope>
</dependency>

<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.15</version>
  <exclusions>
    <exclusion>
      <groupId>javax.mail</groupId>
      <artifactId>mail</artifactId>
    </exclusion>
    <exclusion>
      <groupId>javax.jms</groupId>
      <artifactId>jms</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jdmk</groupId>
      <artifactId>jmxtools</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jmx</groupId>
      <artifactId>jmxri</artifactId>
    </exclusion>
  </exclusions>
  <scope>runtime</scope>
</dependency>

<!-- @Inject -->
<dependency>
  <groupId>javax.inject</groupId>
  <artifactId>javax.inject</artifactId>
  <version>1</version>
</dependency>

<!-- Servlet -->
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
  <scope>provided</scope>
</dependency>
```

```
<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <version>2.1</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>

<!-- Test -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.7</version>
  <scope>test</scope>
</dependency>

<!-- MySQL database driver -->
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.9</version>
</dependency>

<dependency>
  <groupId>commons-dbcp</groupId>
  <artifactId>commons-dbcp</artifactId>
  <version>1.4</version>
</dependency>

<!--iBatis Framework -->
<dependency>
  <groupId>org.apache.ibatis</groupId>
  <artifactId>ibatis-sqlmap</artifactId>
  <version>2.3.4.726</version>
</dependency>

<!-- Tiles dependencies -->
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-api</artifactId>
  <version>2.2.2</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-autotag-core-runtime</artifactId>
  <version>1.1.0</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-core</artifactId>
  <version>2.2.2</version>
</dependency>
```

```
<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-jsp</artifactId>
  <version>2.2.2</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-servlet</artifactId>
  <version>2.2.2</version>
</dependency>

<dependency>
  <groupId>org.apache.tiles</groupId>
  <artifactId>tiles-template</artifactId>
  <version>2.2.2</version>
</dependency>

</dependencies>

Web.xml:

<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
  http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

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

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

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

  <servlet-mapping>
    <servlet-name>appServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>

</web-app>
```

Servlet-context.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc.xsd
        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">
    <annotation-driven />
    <context:property-placeholder location="classpath:jdbc.properties" />
    <beans:bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <beans:property name="prefix" value="/WEB-INF/views/" />
        <beans:property name="suffix" value=".jsp" />
    </beans:bean>

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

    <beans:bean id="userDao" class="com.beingjavaguys.dao.UserDaoImpl" />
    <beans:bean id="userService"
class="com.beingjavaguys.services.UserServiceImpl" />

    <beans:bean
id="propertyConfigurer" class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
        <beans:property name="locations">
            <beans:list>
                <beans:value>classpath:jdbc.properties</beans:value>
            </beans:list>
        </beans:property>
    </beans:bean>

    <beans:bean id="dataSource1"
class="org.apache.commons.dbcp.BasicDataSource">
        <destroy-method>"close"</destroy-method>
        <beans:property name="driverClassName"
value="${app.jdbc.driverClassName}" />
        <beans:property name="url" value="${app.jdbc.url}" />
        <beans:property name="username" value="${app.jdbc.username}" />
        <beans:property name="password" value="${app.jdbc.password}" />
    </beans:bean>

    <beans:bean id="sqlMapClient1"
class="org.springframework.orm.ibatis.SqlMapClientFactoryBean">
        <beans:property name="configLocation" value="classpath:sql-maps-config.xml" />
        <beans:property name="dataSource" ref="dataSource1" />
    </beans:bean>

    <beans:bean id="transactionManager1"
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <beans:property name="dataSource" ref="dataSource1" />
    </beans:bean>
</beans:beans>
```

Jdbc.properties:

```
app.jdbc.driverClassName=com.mysql.jdbc.Driver
app.jdbc.url=jdbc:mysql://localhost:3306/springdb
app.jdbc.username=root
app.jdbc.password=
```

sql-maps-config.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE sqlMapConfig
    PUBLIC "-//ibatis.apache.org//DTD SQL Map Config 2.0//EN"
    "http://ibatis.apache.org/dtd/sql-map-config-2.dtd">
<sqlMapConfig>
    <settings useStatementNamespaces="true"/>
    <transactionManager type="JDBC">
        <dataSource type="SIMPLE">
            <property name="JDBC.Driver" value="com.mysql.jdbc.Driver"/>
            <property name="JDBC.ConnectionURL"
                value="jdbc:mysql://localhost:3306/hibernate"/>
            <property name="JDBC.Username" value="root"/>
            <property name="JDBC.Password" value=""/>
        </dataSource>
    </transactionManager>
    <sqlMap resource="user.xml"/>
</sqlMapConfig>
```

User.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sqlMap PUBLIC "-//ibatis.apache.org//DTD SQL Map 2.0//EN"
"http://ibatis.apache.org/dtd/sql-map-2.dtd">
<sqlMap namespace="user">

    <typeAlias alias="User" type="com.beingjavaguys.domain.User" />

    <resultMap id="userResultMap" class="User">
        <result property="userId" column="user_id" />
        <result property="firstName" column="first_name" />
        <result property="lastName" column="last_name" />
        <result property="gender" column="gender" />
        <result property="city" column="city" />
    </resultMap>

    <select id="listUsers" resultMap="userResultMap">
        SELECT * FROM user
    </select>

    <insert id="insertUser" parameterClass="User">
        INSERT INTO USER
        (FIRST_NAME, LAST_NAME, GENDER, CITY)
        VALUES(#firstName#, #lastName#, #gender#, #city#);
    </insert>

    <delete id="deleteUser" parameterClass="java.lang.Long">
        DELETE FROM USER WHERE USER_ID = #value#
    </delete>
```

```
<select id="getUserById" parameterClass="java.Lang.Long"
        resultMap="userResultMap">
    SELECT * FROM USER WHERE USER_ID = #value#
</select>

<update id="updateUser" parameterClass="User">
    UPDATE USER
    SET FIRST_NAME    = #firstName#,
    LAST_NAME        = #lastName#,
    GENDER            = #gender#,
    CITY              = #city#
    WHERE USER_ID = #userId#
</update>
</sqlMap>
```

Log4j.xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration debug="true"
    xmlns:log4j='http://jakarta.apache.org/Log4j/'>

    <!-- Console -->
    <appender name="console" class="org.apache.Log4j.ConsoleAppender">
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <!-- file -->
    <appender name="file" class="org.apache.Log4j.RollingFileAppender">
        <param name="append" value="false" />
        <param name="maxFileSize" value="10KB" />
        <param name="maxBackupIndex" value="5" />
        <param name="file" value="F:/Coding/SpringCoding/WAE/Logs/Logs.Log" />
        <layout class="org.apache.Log4j.PatternLayout">
            <param name="ConversionPattern"
                value="%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n" />
        </layout>
    </appender>

    <root>
        <level value="DEBUG" />
        <appender-ref ref="console" />
        <appender-ref ref="file" />
    </root>
</log4j:configuration>
```


User.java:

```
package com.beingjavaguys.domain;

public class User {

    private int userId;
    private String firstName;
    private String lastName;
    private String gender;
    private String city;

    public int getUserId() {
        return userId;
    }

    public void setUserId(int userId) {
        this.userId = userId;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
    }

    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }
}
```

HomeController.java:

```
package com.beingjavaguys.controller;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
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.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;
import com.beingjavaguys.domain.User;
import com.beingjavaguys.services.UserService;

@Controller
public class HomeController {

    @Autowired
    UserService userService;

    @RequestMapping("/addregister")
    public ModelAndView getRegisterForm(@ModelAttribute("user") User user,
        BindingResult result) {
        return new ModelAndView("register");
    }

    @RequestMapping("/register")
    public ModelAndView registerUser(@ModelAttribute User user,
        BindingResult result) {

        List<String> genderList = new ArrayList<String>();
        genderList.add("male");
        genderList.add("female");

        List<String> cityList = new ArrayList<String>();
        cityList.add("delhi");
        cityList.add("gurgaon");
        cityList.add("meerut");
        cityList.add("noida");

        Map<String, List> map = new HashMap<String, List>();
        map.put("genderList", genderList);
        map.put("cityList", cityList);
        return new ModelAndView("register", "map", map);
    }

    @RequestMapping("/insert")
    public String insertData(@ModelAttribute User user) {
        if (user != null)
            userService.insertData(user);
        return "redirect:/getList";
    }
}
```

```
@RequestMapping("/getList")
public ModelAndView getUserList() {
    List<User> userList = userService.getUserList();
    return new ModelAndView("userList", "userList", userList);
}

@RequestMapping("/edit")
public ModelAndView editUser(@RequestParam Long id,
    @ModelAttribute User user) {

    user = userService.getUser(id);

    List<String> genderList = new ArrayList<String>();
    genderList.add("male");
    genderList.add("female");

    List<String> cityList = new ArrayList<String>();
    cityList.add("delhi");
    cityList.add("gurgaon");
    cityList.add("meerut");
    cityList.add("noida");

    Map<String, Object> map = new HashMap<String, Object>();
    map.put("genderList", genderList);
    map.put("cityList", cityList);
    map.put("user", user);

    return new ModelAndView("edit", "map", map);
}

@RequestMapping("/update")
public String updateUser(@ModelAttribute User user) {
    userService.updateData(user);
    return "redirect:/getList";
}

@RequestMapping("/delete")
public String deleteUser(@RequestParam Long id) {
    System.out.println("id = " + id);
    userService.deleteData(id);
    return "redirect:/getList";
}

@RequestMapping(value = "/", method = RequestMethod.GET)
public String home(Locale locale, Model model) {
    return "home";
}
}
```

UserService.java:

```
package com.beingjavaguys.services;

import java.util.List;
import com.beingjavaguys.domain.User;

public interface UserService {

    public void insertData(User user);
    public List<User> getUserList();
    public void deleteData(Long id);
    public User getUser(Long id);
    public void updateData(User user);

}
```

UserServiceImpl.java:

```
package com.beingjavaguys.services;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import com.beingjavaguys.dao.UserDao;
import com.beingjavaguys.domain.User;

public class UserServiceImpl implements UserService {

    @Autowired
    UserDao userdao;

    @Override
    public void insertData(User user) {
        userdao.insertData(user);
    }

    @Override
    public List<User> getUserList() {
        return userdao.getUserList();
    }

    @Override
    public void deleteData(Long id) {
        userdao.deleteData(id);
    }

    @Override
    public User getUser(Long id) {
        return userdao.getUser(id);
    }

    @Override
    public void updateData(User user) {
        userdao.updateData(user);
    }

}
```

UserDao.java:

```
package com.beingjavaguys.dao;

import java.util.List;
import com.beingjavaguys.domain.User;

public interface UserDao {
    public Long insertData(User user);
    public List<User> getUserList();
    public void updateData(User user);
    public void deleteData(Long id);
    public User getUser(Long id);
}
```

UserDaoImpl.java:

```
package com.beingjavaguys.dao;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.orm.ibatis.support.SqlMapClientDaoSupport;
import org.springframework.stereotype.Repository;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.beingjavaguys.domain.User;
import com.beingjavaguys.dao.UserDao;

@Repository
public class UserDaoImpl extends SqlMapClientDaoSupport implements UserDao {
    private static final String NAMESPACE = "user.";

    @Autowired
    @Qualifier("sqlMapClient1")
    public void injectSqlMapClient(SqlMapClient sqlMapClient) {
        setSqlMapClient(sqlMapClient);
    }

    @Override
    @SuppressWarnings("unchecked")
    public List<User> getUserList() {
        return getSqlMapClientTemplate().queryForList(NAMESPACE + "listUsers");
    }

    @Override
    public Long insertData(User user) {
        return (Long) getSqlMapClientTemplate().insert(
            NAMESPACE + "insertUser", user);
    }

    @Override
    public void updateData(User user) {
        getSqlMapClientTemplate().update(NAMESPACE + "updateUser", user);
    }
}
```

```
@Override
public void deleteData(Long id) {
    getSqlMapClientTemplate().delete(NAMESPACE + "deleteUser", id);
}

@Override
public User getUser(Long id) {
    return (User) getSqlMapClientTemplate().queryForObject(
        NAMESPACE + "getUserById", id);
}
}
```

Registration Form

First Name :

Last Name :

Gender : ☒ male ☐ female

City :

[Click Here to See User List](#)

Edit User Details

First Name :

Last Name :

Gender : ☒ male ☐ female

City :

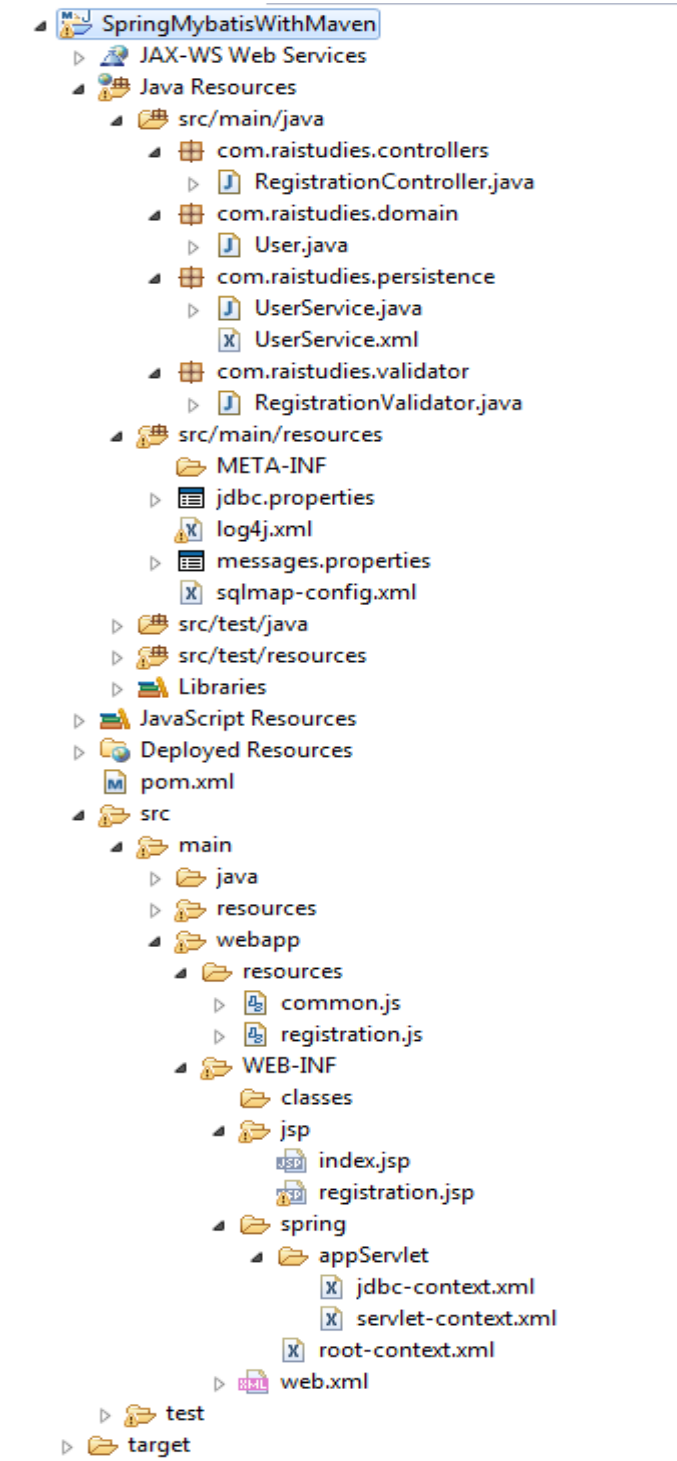
User List						
User Id	First Name	Last Name	Gender	City	Edit	Delete
1	Amit	Bhalerao	male	delhi	Edit	Delete
2	Kasturi Amit	Bhalerao	female	noida	Edit	Delete
3	Amey	Bhalerao	male	meerut	Edit	Delete
6	Avani	Bhalerao	female	gurgaon	Edit	Delete
9	Test	Test	male	gurgaon	Edit	Delete
Add New User						

User List						
User Id	First Name	Last Name	Gender	City	Edit	Delete
1	Amit	Bhalerao	male	delhi	Edit	Delete
2	Kasturi Amit	Bhalerao	female	noida	Edit	Delete
3	Amey	Bhalerao	male	meerut	Edit	Delete
6	Avani	Bhalerao	female	gurgaon	Edit	Delete
9	Test11111	Test111	male	meerut	Edit	Delete
Add New User						

➤ Spring MyBatis Integration

1) Using XML Mapping Configuration

Project Structure :



Pom.xml:

```
<dependency>
  <groupId>aopalliance</groupId>
  <artifactId>aopalliance</artifactId>
  <version>1.0</version>
</dependency>

<dependency>
  <groupId>c3p0</groupId>
  <artifactId>c3p0</artifactId>
  <version>0.9.1.2</version>
</dependency>

<dependency>
  <groupId>commons-logging</groupId>
  <artifactId>commons-logging</artifactId>
  <version>1.1.1</version>
</dependency>

<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis-spring</artifactId>
  <version>1.2.2</version>
</dependency>

<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis</artifactId>
  <version>3.2.7</version>
</dependency>

<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.21</version>
</dependency>

<!-- Servlet -->
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
  <scope>provided</scope>
</dependency>

<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <version>2.1</version>
  <scope>provided</scope>
</dependency>

<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```



```
<!-- AspectJ -->
<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
</dependency>

<!-- @Inject -->
<dependency>
    <groupId>javax.inject</groupId>
    <artifactId>javax.inject</artifactId>
    <version>1</version>
</dependency>

<!-- Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-aop</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-asm</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>org.springframework.beans</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${org.springframework-version}</version>
    <exclusions>
        <!-- Exclude Commons Logging in favor of SLF4j -->
        <exclusion>
            <groupId>commons-logging</groupId>
            <artifactId>commons-logging</artifactId>
        </exclusion>
    </exclusions>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-support</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>${org.springframework-version}</version>
</dependency>
```

```
<!-- Jdbc -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>org.springframework.expression</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-orm</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<!-- AspectJ -->
<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
</dependency>

<!-- Logging -->
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-api</artifactId>
    <version>${org.slf4j-version}</version>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>jcl-over-slf4j</artifactId>
    <version>${org.slf4j-version}</version>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-log4j12</artifactId>
    <version>${org.slf4j-version}</version>
    <scope>runtime</scope>
</dependency>
```

```
<dependency>
  <groupId>log4j</groupId>
  <artifactId>log4j</artifactId>
  <version>1.2.15</version>
  <exclusions>
    <exclusion>
      <groupId>javax.mail</groupId>
      <artifactId>mail</artifactId>
    </exclusion>
    <exclusion>
      <groupId>javax.jms</groupId>
      <artifactId>jms</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jdmk</groupId>
      <artifactId>jmxtools</artifactId>
    </exclusion>
    <exclusion>
      <groupId>com.sun.jmx</groupId>
      <artifactId>jmxri</artifactId>
    </exclusion>
  </exclusions>
  <scope>runtime</scope>
</dependency>

<!-- @Inject -->
<dependency>
  <groupId>javax.inject</groupId>
  <artifactId>javax.inject</artifactId>
  <version>1</version>
</dependency>

<!-- Test -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.7</version>
  <scope>test</scope>
</dependency>
```

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"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
  version="2.5">
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/root-context.xml</param-value>
  </context-param>
  <listener>
    <listener-class>
      org.springframework.web.context.ContextLoaderListener
    </listener-class>
  </listener>
```

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

Servlet-context.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"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    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-3.0.xsd
        http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd">

    <mvc:resources mapping="/resources/**" location="/resources/" />

    <!-- Application Message Bundle -->
    <bean id="messageSource"
class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
        <property name="basename" value="classpath:messages" />
        <property name="cacheSeconds" value="3000" />
    </bean>

    <!-- Scans the classpath of this application for @Components to deploy as
beans -->
    <context:component-scan base-package="com.raistudies" />

    <!-- Configures the @Controller programming model -->
    <mvc:annotation-driven />

    <!-- Resolves view names to protected .jsp resources within the /WEB-
INF/views directory -->
    <bean id="viewResolver"
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/jsp/" />
        <property name="suffix" value=".jsp" />
    </bean>

    <import resource="jdbc-context.xml" />
</beans>
```

jdbc-context.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:p="http://www.springframework.org/schema/p"
       xmlns:tx="http://www.springframework.org/schema/tx"
       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/tx
           http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context-
3.0.xsd
       ">

    <context:property-placeholder location="classpath:jdbc.properties" />

    <!-- Enable annotation style of managing transactions -->
    <tx:annotation-driven transaction-manager="transactionManager" />

    <!-- Declare a datasource that has pooling capabilities -->
    <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"
          destroy-method="close" p:driverClass="${app.jdbc.driverClassName}"
          p:jdbcUrl="${app.jdbc.url}" p:user="${app.jdbc.username}"
p:password="${app.jdbc.password}"
          p:acquireIncrement="10" p:idleConnectionTestPeriod="60"
p:maxPoolSize="100"
          p:maxStatements="50" p:minPoolSize="10" />

    <!-- Declare a transaction manager -->
    <bean id="transactionManager"

        class="org.springframework.jdbc.datasource.DataSourceTransactionManager"
        p:dataSource-ref="dataSource" />

    <!-- define the SqlSessionFactory, notice that configLocation is not needed
when you use MapperFactoryBean -->
    <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
        <property name="dataSource" ref="dataSource" />
        <property name="configLocation" value="classpath:sqlmap-config.xml" />
    </bean>

    <!-- scan for mappers and let them be autowired -->
    <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
        <property name="basePackage" value="com.raistudies.persistence" />
    </bean>

</beans>
```

Jdbc.properties:

```
app.jdbc.driverClassName=com.mysql.jdbc.Driver
app.jdbc.url=jdbc:mysql://localhost/hibernate
app.jdbc.username=root
app.jdbc.password=
```

messages.properties:

field.name.empty=Name field is mandatory.
field.standard.empty=Standard field is mandatory.
field.age.empty=Age field is mandatory.
field.sex.empty=Sex field is mandatory.
field.age.NAN=Age should be a number.

sqlmap-config.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
    PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>
    <settings>
        <!-- changes from the defaults -->
        <setting name="lazyLoadingEnabled" value="false" />
    </settings>
    <typeAliases>
        <typeAlias type="com.raistudies.domain.User" alias="user"/>
    </typeAliases>
</configuration>
```

UserService.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="com.raistudies.persistence.UserService">

    <resultMap id="result" type="user">
        <result property="id" column="id"/>
        <result property="name" column="name"/>
        <result property="standard" column="standard"/>
        <result property="age" column="age"/>
        <result property="sex" column="sex"/>
    </resultMap>

    <select id="getAllUser" resultMap="result">
        SELECT id,name,standard,age,sex
        FROM user;
    </select>

    <insert id="saveUser" parameterType="user">
        INSERT INTO user (name,standard,age,sex)
        VALUE (#{name},#{standard},#{age},#{sex})
    </insert>

    <update id="updateUser" parameterType="user">
        UPDATE user
        SET
            name = #{name},
            standard = #{standard},
            age = #{age},
```

```
        sex = #{sex}
    where id = #{id}
</update>

<delete id="deleteUser" parameterType="string">
    DELETE FROM user
    WHERE id = #{id}
</delete>
</mapper>
```

User.java:

```
package com.raistudies.domain;

import java.io.Serializable;

public class User implements Serializable {

    private static final long serialVersionUID = 3647233284813657927L;

    private String id;
    private String name = null;
    private String standard = null;
    private String age;
    private String sex = null;

    public String getId() {
        return id;
    }

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

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getStandard() {
        return standard;
    }

    public void setStandard(String standard) {
        this.standard = standard;
    }

    public String getAge() {
        return age;
    }

    public void setAge(String age) {
        this.age = age;
    }

    public String getSex() {
```

```
        return sex;
    }

    public void setSex(String sex) {
        this.sex = sex;
    }

    @Override
    public String toString() {
        return "User [name=" + name + ", standard=" + standard + ", age=" + age
            + ", sex=" + sex + "]";
    }
}
```

RegistrationController.java:

```
package com.raistudies.controllers;

import java.util.List;
import java.util.UUID;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
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 com.raistudies.domain.User;
import com.raistudies.persistence.UserService;
import com.raistudies.validator.RegistrationValidator;

@Controller
@RequestMapping(value="/registration")
public class RegistrationController {

    private RegistrationValidator validator = null;
    private UserService userService = null;

    @Autowired
    public void setUserService(UserService userService) {
        this.userService = userService;
    }

    public RegistrationValidator getValidator() {
        return validator;
    }

    @Autowired
    public void setValidator(RegistrationValidator validator) {
        this.validator = validator;
    }

    @RequestMapping(method=RequestMethod.GET)
    public String showForm(ModelMap model){
        List<User> users = userService.getAllUser();
        model.addAttribute("users", users);
        User user = new User();
    }
}
```



```
        user.setId(UUID.randomUUID().toString());
        model.addAttribute("user", user);
        return "registration";
    }

    @RequestMapping(value="/add", method=RequestMethod.POST)
    public ModelAndView add(@ModelAttribute(value="user") User
user, BindingResult result){
        validator.validate(user, result);
        ModelAndView mv = new ModelAndView("registration");
        if(!result.hasErrors()){
            userService.saveUser(user);
            user = new User();
            user.setId(UUID.randomUUID().toString());
            mv.addObject("user", user);
        }
        mv.addObject("users", userService.getAllUser());
        return mv;
    }

    @RequestMapping(value="/update", method=RequestMethod.POST)
    public ModelAndView update(@ModelAttribute(value="user") User
user, BindingResult result){
        validator.validate(user, result);
        ModelAndView mv = new ModelAndView("registration");
        if(!result.hasErrors()){
            userService.updateUser(user);
            user = new User();
            user.setId(UUID.randomUUID().toString());
            mv.addObject("user", user);
        }
        mv.addObject("users", userService.getAllUser());
        return mv;
    }

    @RequestMapping(value="/delete", method=RequestMethod.POST)
    public ModelAndView delete(@ModelAttribute(value="user") User
user, BindingResult result){
        validator.validate(user, result);
        ModelAndView mv = new ModelAndView("registration");
        if(!result.hasErrors()){
            userService.deleteUser(user.getId());
            user = new User();
            user.setId(UUID.randomUUID().toString());
            mv.addObject("user", user);
        }
        mv.addObject("users", userService.getAllUser());
        return mv;
    }
}
```

RegistrationValidator.java:

```
package com.raistudies.validator;

import org.springframework.stereotype.Component;
import org.springframework.validation.Errors;
import org.springframework.validation.ValidationUtils;
import org.springframework.validation.Validator;
import com.raistudies.domain.User;

@Component
public class RegistrationValidator implements Validator {

    public boolean supports(Class<?> c) {
        return User.class.isAssignableFrom(c);
    }

    public void validate(Object command, Errors errors) {
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "name",
            "field.name.empty");
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "standard",
            "field.standard.empty");
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "age",
            "field.age.empty");
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "sex",
            "field.sex.empty");
        User usrBean = (User) command;
        if (!isNumber(usrBean.getAge().trim()))
            errors.rejectValue("age", "field.age.NAN");
    }

    private boolean isNumber(String str) {
        for (int i = 0; i < str.length(); i++) {

            // If we find a non-digit character we return false.
            if (!Character.isDigit(str.charAt(i)))
                return false;
        }

        return true;
    }
}
```

UserService.java:

```
package com.raistudies.persistance;

import java.util.List;
import com.raistudies.domain.User;

public interface UserService {
    public void saveUser(User user);
    public void updateUser(User user);
    public void deleteUser(String id);
    public List<User> getAllUser();
}
```

Registration.jsp:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<%@ page session="true" %>
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Spring MyBatis Integration</title>
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
    <script type="text/javascript" src="<c:url
value='/resources/common.js'/>"></script>
    <script type="text/javascript" src="<c:url
value='/resources/registration.js'/>"></script>
    <script type="text/javascript">
      var projectUrl = '<c:url value="/" />';
      if(projectUrl.indexOf("; ", 0) != -1){
        projectUrl = projectUrl.substring(0,
        projectUrl.indexOf("; ", 0));
      }
    </script>
  </head>
  <body>
    <fieldset>
      <legend>Registration Form</legend>
      <center>
        <form:form commandName="user"
action="/SpringMybatisWithMaven/registration/add" name="userForm">

        <form:hidden path="id"/>
        <table>
          <tr><td colspan="2" align="left"><form:errors path="*"
cssStyle="color : red;"/></td></tr>
          <tr><td>Name : </td><td><form:input path="name" /></td></tr>
          <tr><td>Standard : </td><td><form:input path="standard"
/></td></tr>
          <tr><td>Age : </td><td><form:input path="age" /></td></tr>
          <tr><td>Sex : </td><td><form:select path="sex">
            <form:option value="Male"/>
            <form:option value="Female"/>
          </form:select></td></tr>
          <tr><td colspan="2"><input type="submit" value="Save
Changes"/>
          &nbsp;<input type="reset" name="newUser" value="New User"
onclick="setAddForm();" disabled="disabled"/>
          &nbsp;<input type="submit" name="deleteUser" value="Delete
User" onclick="setDeleteForm();"
disabled="disabled"/></td></tr>
        </table>
      </form:form>
    </center>
  </fieldset>
</body>
</html>
```

```
<c:if test="${!empty users}">
<br />
    <center>
        <table width="90%">
            <tr style="background-color: gray;">
                <th>Name</th>
                <th>Standard</th>
                <th>Age</th>
                <th>Sex</th>
            </tr>
            <c:forEach items="${users}" var="user">
                <tr style="background-color: silver;"
                    id="${user.id}" onclick="setUpdateForm('${user.id}');">
                    <td><c:out value="${user.name}"/></td>
                    <td><c:out value="${user.standard}"/></td>
                    <td><c:out value="${user.age}"/></td>
                    <td><c:out value="${user.sex}"/></td>
                </tr>
            </c:forEach>
        </table>
    </center>
<br />
</c:if>
</body>
</html>
```

Common.js:

```
function changeSelectValue(selectNode, value){
for(var index=0;index<selectNode.length;index++) {
    if(selectNode[index].value == value)
        selectNode.selectedIndex = index;
}
}
```

Registration.js:

```
var newUserId;

var addUrl = "/registration/add";

var deleteUrl = "/registration/delete";

var updateUrl = "/registration/update";
```

```
function setUpdateForm(element){

    newUserId = document.userForm.id.value;

    trElement = document.getElementById(element);

    document.userForm.id.value = element;

    document.userForm.name.value =
trElement.children[0].innerHTML;

    document.userForm.standard.value =
trElement.children[1].innerHTML;

    document.userForm.age.value =
trElement.children[2].innerHTML;

    changeSelectValue(document.userForm.sex,
trElement.children[3].innerHTML);

    document.userForm.newUser.disabled = false;

    document.userForm.deleteUser.disabled = false;

    document.userForm.action = projectUrl + updateUrl;

}

function setAddForm(){

    document.userForm.id.value = newUserId;

    document.userForm.name.value = "";

    document.userForm.standard.value = "";

    document.userForm.age.value = "";

    changeSelectValue(document.userForm.sex, "Male");

    document.userForm.newUser.disabled = true;

    document.userForm.deleteUser.disabled = true;

    document.userForm.action = projectUrl + addUrl;

}

function setDeleteForm() {

    document.userForm.action = projectUrl + deleteUrl;

    return true;

}
```

Registration Form

Name :

Standard :

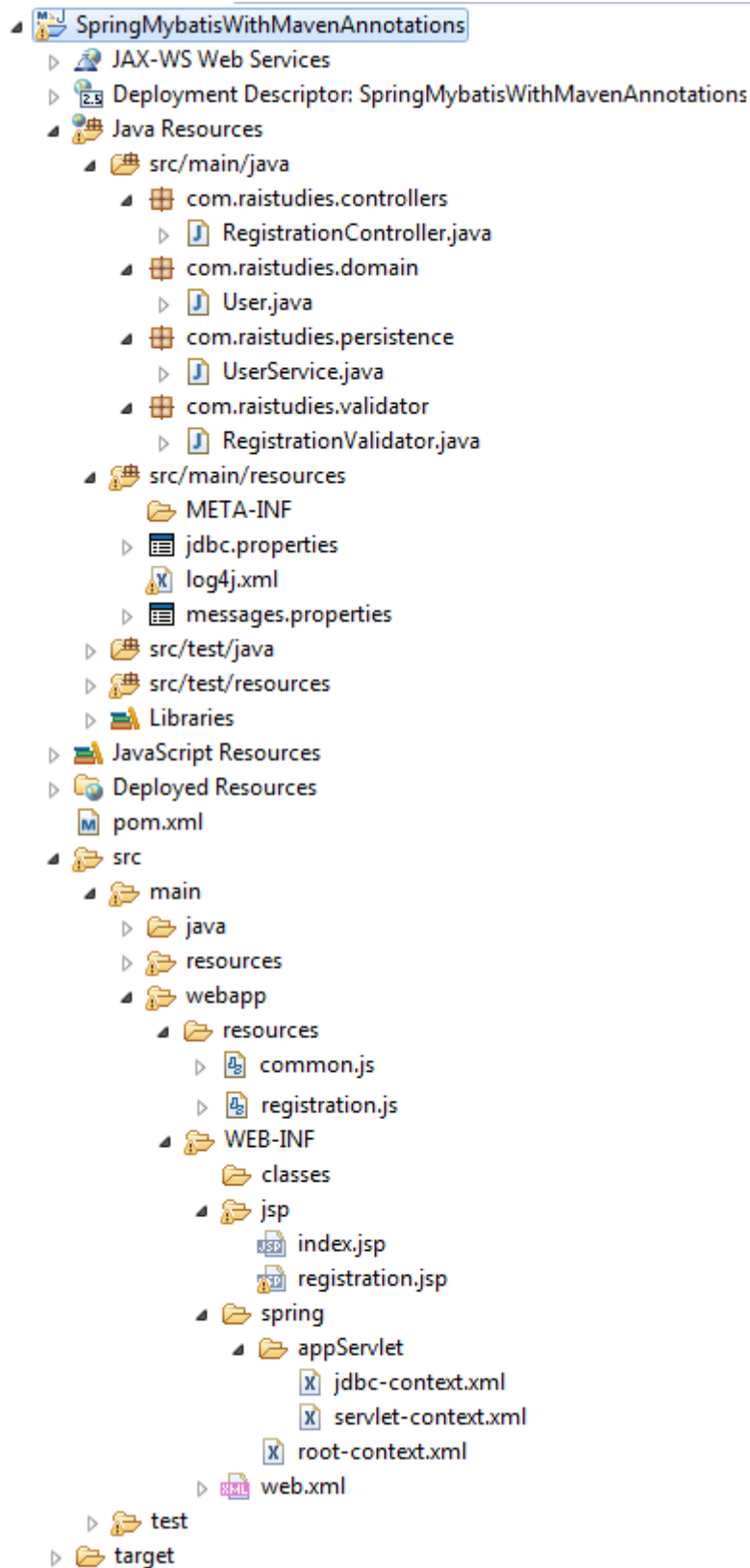
Age :

Sex :

Name	Standard	Age	Sex
Amit Anil Bhalerao	MCA	29	Male
Mrs Kasturi Amit Bhalerao	MCS	28	Male

2) Using Annotations

Project Structure:



Pom.xml:

```
<dependency>
  <groupId>aopalliance</groupId>
  <artifactId>aopalliance</artifactId>
  <version>1.0</version>
</dependency>

<dependency>
  <groupId>c3p0</groupId>
  <artifactId>c3p0</artifactId>
  <version>0.9.1.2</version>
</dependency>

<dependency>
  <groupId>commons-logging</groupId>
  <artifactId>commons-logging</artifactId>
  <version>1.1.1</version>
</dependency>

<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis-spring</artifactId>
  <version>1.2.2</version>
</dependency>

<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis</artifactId>
  <version>3.2.7</version>
</dependency>

<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>5.1.21</version>
</dependency>

<!-- Servlet -->
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
  <scope>provided</scope>
</dependency>

<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <version>2.1</version>
  <scope>provided</scope>
</dependency>

<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```



```
<!-- AspectJ -->
<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
</dependency>

<!-- @Inject -->
<dependency>
    <groupId>javax.inject</groupId>
    <artifactId>javax.inject</artifactId>
    <version>1</version>
</dependency>

<!-- Spring -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-aop</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-asm</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>org.springframework.beans</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${org.springframework-version}</version>
    <exclusions>
        <!-- Exclude Commons Logging in favor of SLF4j -->
        <exclusion>
            <groupId>commons-logging</groupId>
            <artifactId>commons-logging</artifactId>
        </exclusion>
    </exclusions>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-support</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>${org.springframework-version}</version>
</dependency>
```

```
<!-- Jdbc -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>org.springframework.expression</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-orm</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>${org.springframework-version}</version>
</dependency>

<!-- AspectJ -->
<dependency>
    <groupId>org.aspectj</groupId>
    <artifactId>aspectjrt</artifactId>
    <version>${org.aspectj-version}</version>
</dependency>

<!-- Logging -->
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-api</artifactId>
    <version>${org.slf4j-version}</version>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>jcl-over-slf4j</artifactId>
    <version>${org.slf4j-version}</version>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-log4j12</artifactId>
    <version>${org.slf4j-version}</version>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.15</version>
    <exclusions>
        <exclusion>
            <groupId>javax.mail</groupId>
            <artifactId>mail</artifactId>
        </exclusion>
    </exclusions>
</dependency>
```

```
        <exclusion>
            <groupId>javax.jms</groupId>
            <artifactId>jms</artifactId>
        </exclusion>
        <exclusion>
            <groupId>com.sun.jdmk</groupId>
            <artifactId>jmxtools</artifactId>
        </exclusion>
        <exclusion>
            <groupId>com.sun.jmx</groupId>
            <artifactId>jmxri</artifactId>
        </exclusion>
    </exclusions>
    <scope>runtime</scope>
</dependency>

<!-- @Inject -->
<dependency>
    <groupId>javax.inject</groupId>
    <artifactId>javax.inject</artifactId>
    <version>1</version>
</dependency>

<!-- Test -->
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
</dependency>
```

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"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
    version="2.5">
    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/spring/root-context.xml</param-value>
    </context-param>
    <listener>
        <listener-class>
            org.springframework.web.context.ContextLoaderListener
        </listener-class>
    </listener>
    <servlet>
        <servlet-name>appServlet</servlet-name>
        <servlet-class>
            org.springframework.web.servlet.DispatcherServlet
        </servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>
                /WEB-INF/spring/appServlet/<u>servlet</u>-context.xml
            </param-value>
        </init-param>
```

```
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>appServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
</web-app>
```

Servlet-context.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"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       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-3.0.xsd
           http://www.springframework.org/schema/mvc
           http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd">

    <mvc:resources mapping="/resources/**" location="/resources/" />

    <!-- Application Message Bundle -->
    <bean id="messageSource"
class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
        <property name="basename" value="classpath:messages" />
        <property name="cacheSeconds" value="3000" />
    </bean>

    <!-- Scans the classpath of this application for @Components to deploy as
beans -->
    <context:component-scan base-package="com.raistudies" />

    <!-- Configures the @Controller programming model -->
    <mvc:annotation-driven />

    <!-- Resolves view names to protected .jsp resources within the /WEB-
INF/views directory -->
    <bean id="viewResolver"
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/jsp/" />
        <property name="suffix" value=".jsp" />
    </bean>

    <import resource="jdbc-context.xml"/>
</beans>
```

Jdbc-context.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:p="http://www.springframework.org/schema/p"
       xmlns:tx="http://www.springframework.org/schema/tx"
       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/tx
           http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context-
3.0.xsd
       ">

    <context:property-placeholder location="classpath:jdbc.properties" />

    <!-- Enable annotation style of managing transactions -->
    <tx:annotation-driven transaction-manager="transactionManager" />

    <!-- Declare a datasource that has pooling capabilities -->
    <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"
          destroy-method="close" p:driverClass="${app.jdbc.driverClassName}"
          p:jdbcUrl="${app.jdbc.url}" p:user="${app.jdbc.username}"
          p:password="${app.jdbc.password}"
          p:acquireIncrement="10" p:idleConnectionTestPeriod="60"
          p:maxPoolSize="100"
          p:maxStatements="50" p:minPoolSize="10" />

    <!-- Declare a transaction manager -->
    <bean id="transactionManager"

        class="org.springframework.jdbc.datasource.DataSourceTransactionManager"
        p:dataSource-ref="dataSource" />

    <!-- define the SqlSessionFactory, notice that configLocation is not needed
    when you use MapperFactoryBean -->
    <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
        <property name="dataSource" ref="dataSource" />
    </bean>

    <!-- scan for mappers and let them be autowired -->
    <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
        <property name="basePackage" value="com.raistudies.persistence" />
    </bean>

</beans>
```

Jdbc.properties:

```
app.jdbc.driverClassName=com.mysql.jdbc.Driver
app.jdbc.url=jdbc:mysql://localhost/hibernate
app.jdbc.username=root
app.jdbc.password=
```

messages.properties:

```
field.name.empty=Name field is mandatory.
field.standard.empty=Standard field is mandatory.
field.age.empty=Age field is mandatory.
field.sex.empty=Sex field is mandatory.
field.age.NAN=Age should be a number.
```

User.java:

```
package com.raistudies.domain;

import java.io.Serializable;

public class User implements Serializable {

    private static final long serialVersionUID = 3647233284813657927L;

    private String id;
    private String name = null;
    private String standard = null;
    private String age;
    private String sex = null;

    public String getId() {
        return id;
    }

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

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getStandard() {
        return standard;
    }

    public void setStandard(String standard) {
        this.standard = standard;
    }

    public String getAge() {
        return age;
    }
}
```

```
    public void setAge(String age) {
        this.age = age;
    }

    public String getSex() {
        return sex;
    }

    public void setSex(String sex) {
        this.sex = sex;
    }

    @Override
    public String toString() {
        return "User [name=" + name + ", standard=" + standard + ", age=" + age
            + ", sex=" + sex + "]";
    }
}
```

RegistrationController.java:

```
package com.raistudies.controllers;

import java.text.DateFormat;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import java.util.UUID;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
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 com.raistudies.domain.User;
import com.raistudies.persistence.UserService;
import com.raistudies.validator.RegistrationValidator;

@Controller
@RequestMapping(value="/registration")
public class RegistrationController {

    private RegistrationValidator validator = null;
    private UserService userService = null;

    @Autowired
    public void setUserService(UserService userService) {
        this.userService = userService;
    }

    public RegistrationValidator getValidator() {
        return validator;
    }
}
```

```
@Autowired
public void setValidator(RegistrationValidator validator) {
    this.validator = validator;
}

@RequestMapping(value = "/", method = RequestMethod.GET)
public String home(Locale locale, Model model) {

    Date date = new Date();
    DateFormat dateFormat =
DateFormat.getDateInstance(DateFormat.LONG, DateFormat.LONG, locale);

    String formattedDate = dateFormat.format(date);

    model.addAttribute("serverTime", formattedDate );

    return "/registration";
}

@RequestMapping(method=RequestMethod.GET)
public String showForm(ModelMap model,@ModelAttribute(value="user") User
user){

    List<User> users = userService.getAllUser();
    model.addAttribute("users", users);
    user.setId(UUID.randomUUID().toString());
    model.addAttribute("user", user);
    return "registration";
}

@RequestMapping(value="/add", method=RequestMethod.POST)
public ModelAndView add(@ModelAttribute(value="user") User
user,BindingResult result){
    validator.validate(user, result);
    ModelAndView mv = new ModelAndView("registration");
    if(!result.hasErrors()){
        userService.saveUser(user);
        user = new User();
        user.setId(UUID.randomUUID().toString());
        mv.addObject("user", user);
    }
    mv.addObject("users", userService.getAllUser());
    return mv;
}

@RequestMapping(value="/update", method=RequestMethod.POST)
public ModelAndView update(@ModelAttribute(value="user") User
user,BindingResult result){
    validator.validate(user, result);
    ModelAndView mv = new ModelAndView("registration");
    if(!result.hasErrors()){
        userService.updateUser(user);
        user = new User();
        user.setId(UUID.randomUUID().toString());
        mv.addObject("user", user);
    }
    mv.addObject("users", userService.getAllUser());
    return mv;
}
```



```
@RequestMapping(value="/delete", method=RequestMethod.POST)
public ModelAndView delete(@ModelAttribute(value="user") User
user, BindingResult result){
    validator.validate(user, result);
    ModelAndView mv = new ModelAndView("registration");
    if(!result.hasErrors()){
        userService.deleteUser(user.getId());
        user = new User();
        user.setId(UUID.randomUUID().toString());
        mv.addObject("user", user);
    }
    mv.addObject("users", userService.getAllUser());
    return mv;
}
```

RegistrationValidator.java:

```
package com.raistudies.validator;

import org.springframework.stereotype.Component;
import org.springframework.validation.Errors;
import org.springframework.validation.ValidationUtils;
import org.springframework.validation.Validator;
import com.raistudies.domain.User;

@Component
public class RegistrationValidator implements Validator {

    public boolean supports(Class<?> c) {
        return User.class.isAssignableFrom(c);
    }

    public void validate(Object command, Errors errors) {
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "name",
            "field.name.empty");
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "standard",
            "field.standard.empty");
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "age",
            "field.age.empty");
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "sex",
            "field.sex.empty");
        User usrBean = (User)command;
        if(!isNumber(usrBean.getAge().trim()))
            errors.rejectValue("age", "field.age.NAN");
    }

    private boolean isNumber(String str){
        for (int i = 0; i < str.length(); i++) {

            //If we find a non-digit character we return false.
            if (!Character.isDigit(str.charAt(i)))
                return false;
        }

        return true;
    }
}
```

UserService.java:

```
package com.raistudies.persistence;

import java.util.List;
import com.raistudies.domain.User;
import org.apache.ibatis.annotations.Insert;
import org.apache.ibatis.annotations.Options;
import org.apache.ibatis.annotations.Select;
import org.apache.ibatis.annotations.Update;
import org.apache.ibatis.annotations.Delete;

public interface UserService {

    @Insert("INSERT INTO user (name,standard,age,sex) VALUES"
           + "({name},{standard},{age},{sex})")
    @Options(useGeneratedKeys = true, keyProperty = "id", flushCache = true,
            keyColumn = "id")
    public void saveUser(User user);

    @Update("UPDATE user SET name = {name},"
           + "standard = {standard}," + "age = {age}, " + "sex = {sex} "
           + "where id = {id}")
    public void updateUser(User user);

    @Delete("DELETE FROM user WHERE id = {id}")
    public void deleteUser(String id);

    @Select("SELECT id,name,standard,age,sex FROM user")
    public List<User> getAllUser();
}
```