

Spring MVC with Database

Steps –

- Create schema in mysql database.
- Create required tables.
- Create new maven project.
- Add all required dependencies in pom.xml
- Create a model class to hold information and store in database.
- Create DAO classes to interact with database and perform some CRUD operations on database.
- Create XML file for spring-mvc configuration.
- Configure database details in configuration file.
- Define all required beans.
- Define Web.xml in WEB-INF directory
- Create controller class to handle requests from frontend.
- Create required JSP files for post and get requests

Key Points-

- Model is an interface while ModelMap is a class.
- ModelAndView is just a container for both a ModelMap and a View object.
- It allows a controller to return both as a single value. I usually like ModelAndView to return the model and view from a controller
- The @ModelAttribute is an annotation that binds a method parameter or method return value to a named model attribute and then exposes it to a web view.

Create database and table

```
create database PersonDatabase;
use PersonDatabase;

CREATE TABLE `Person` (
  `person_id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
```

```
`email` varchar(45) NOT NULL,  
`address` varchar(45) NOT NULL,  
`telephone` varchar(45) NOT NULL,  
PRIMARY KEY (`person_id`)  
) ENGINE=InnoDB AUTO_INCREMENT=25 DEFAULT CHARSET=utf8
```

pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  
http://maven.apache.org/xsd/maven-4.0.0.xsd">  
  <modelVersion>4.0.0</modelVersion>  
  
  <groupId>com.javaTraining</groupId>  
  <artifactId>Spring_MVC_WithJdbc_Example</artifactId>  
  <version>1</version>  
  <packaging>war</packaging>  
  
  <properties>  
    <spring.version>4.0.5.RELEASE</spring.version>  
  </properties>  
  
  <dependencies>  
    <dependency>  
      <groupId>org.springframework</groupId>  
      <artifactId>spring-core</artifactId>  
      <version>${spring.version}</version>  
    </dependency>  
    <dependency>  
      <groupId>org.springframework</groupId>  
      <artifactId>spring-web</artifactId>  
      <version>${spring.version}</version>  
    </dependency>  
    <dependency>  
      <groupId>org.springframework</groupId>  
      <artifactId>spring-webmvc</artifactId>  
      <version>${spring.version}</version>  
    </dependency>  
    <dependency>  
      <groupId>org.springframework</groupId>  
      <artifactId>spring-orm</artifactId>  
      <version>${spring.version}</version>  
      <scope>compile</scope>  
    </dependency>
```

```

<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>javax.servlet-api</artifactId>
  <version>3.1.0</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.11</version>
</dependency>
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
</dependencies>

<!--<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
    </plugin>
  </plugins>
</build>-->

</project>

```

Person class (MODEL)

```

package com.javaTraining;

public class Person {

  private int id;
  private String name;
  private String email;
  private String address;
  private String telephone;

```

```
public Person() {  
}  
  
public Person(String name, String email, String address, String telephone) {  
    this.name = name;  
    this.email = email;  
    this.address = address;  
    this.telephone = telephone;  
}  
  
public int getId() {  
    return id;  
}  
  
public void setId(int id) {  
    this.id = id;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public String getEmail() {  
    return email;  
}  
  
public void setEmail(String email) {  
    this.email = email;  
}  
  
public String getAddress() {  
    return address;  
}  
  
public void setAddress(String address) {  
    this.address = address;  
}  
  
public String getTelephone() {  
    return telephone;  
}
```

```

    public void setTelephone(String telephone) {
        this.telephone = telephone;
    }
}

```

PersonDao (Data Access Object)

```

package com.javaTraining;

import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.ResultSetExtractor;
import org.springframework.jdbc.core.RowMapper;

import javax.sql.DataSource;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;

public class PersonDao {

    private JdbcTemplate jdbcTemplate;

    public PersonDao(DataSource dataSource) {
        jdbcTemplate = new JdbcTemplate(dataSource);
    }

    public void saveOrUpdate(Person person) {
        if (person.getId() > 0) {
            // update
            String sql = "UPDATE contact SET name=?, email=?, address=?, "
                + "telephone=? WHERE contact_id=?";
            jdbcTemplate.update(sql, person.getName(), person.getEmail(),
                person.getAddress(), person.getTelephone(), person.getId());
        } else {
            // insert
            String sql = "INSERT INTO contact (name, email, address, telephone)"
                + " VALUES (?, ?, ?, ?)";
            jdbcTemplate.update(sql, person.getName(), person.getEmail(),
                person.getAddress(), person.getTelephone());
        }
    }

    public void delete(int personId) {
        String sql = "DELETE FROM contact WHERE contact_id=?";
        jdbcTemplate.update(sql, personId);
    }
}

```

```

public Person get(int personId) {
    String sql = "SELECT * FROM contact WHERE contact_id=" + personId;
    return jdbcTemplate.query(sql, new ResultSetExtractor<Person>() {

        public Person extractData(ResultSet rs) throws SQLException,
            DataAccessException {
            if (rs.next()) {
                Person person = new Person();
                person.setId(rs.getInt("contact_id"));
                person.setName(rs.getString("name"));
                person.setEmail(rs.getString("email"));
                person.setAddress(rs.getString("address"));
                person.setTelephone(rs.getString("telephone"));
                return person;
            }
            return null;
        }

    });
}

public List<Person> list() {
    String sql = "SELECT * FROM person";
    List<Person> listPerson = jdbcTemplate.query(sql, new RowMapper<Person>() {

        public Person mapRow(ResultSet rs, int rowNum) throws SQLException {
            Person aPerson = new Person();

            aPerson.setId(rs.getInt("person_id"));
            aPerson.setName(rs.getString("name"));
            aPerson.setEmail(rs.getString("email"));
            aPerson.setAddress(rs.getString("address"));
            aPerson.setTelephone(rs.getString("telephone"));

            return aPerson;
        }

    });

    return listPerson;
}

```

SpringConfig-servlet.xml

```
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:context="http://www.springframework.org/schema/context"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.0.xsd">
    <context:component-scan base-package="com.javaTraining"/>
    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/jsp/" />
        <property name="suffix" value=".jsp" />
    </bean>

    <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
        <property name="url" value="jdbc:mysql://localhost:3306/PersonDatabase"/>
        <property name="username" value="root"/>
        <property name="password" value="12345"/>
    </bean>

    <bean name="personDao" class="com.javaTraining.PersonDao" autowire="byName">
        <constructor-arg ref="dataSource"/>
    </bean>

</beans>
```

Web.xml

```
<web-app id="WebApp_ID" version="2.4"
    xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
    <display-name>Spring MVC Application</display-name>
    <servlet>
        <servlet-name>SpringConfig</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>SpringConfig</servlet-name>
        <url-pattern>/</url-pattern>
```

```
</servlet-mapping>
</web-app>
```

PersonController

```
package com.javaTraining;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
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 javax.servlet.http.HttpServletRequest;
import java.io.IOException;
import java.util.List;

@Controller
public class PersonController {

    @Autowired
    private PersonDao personDao;

    @RequestMapping(value="/",method = RequestMethod.GET)
    public ModelAndView listPerson(ModelAndView model) throws IOException {
        List<Person> listPerson = personDao.list();
        model.addObject("listPerson", listPerson);
        model.setViewName("home");

        return model;
    }

    @RequestMapping(value = "/newContact", method = RequestMethod.GET)
    public ModelAndView newPerson(ModelAndView modelAndView){
        Person newperson = new Person();
        modelAndView.addObject("person", newperson);
        modelAndView.setViewName("PersonForm");
        return modelAndView;
    }

    @RequestMapping(value = "/savePerson", method = RequestMethod.POST)
    public ModelAndView savePerson(@ModelAttribute Person person) {
        personDao.saveOrUpdate(person);
        return new ModelAndView("redirect:/");
    }
}
```



```
}

@RequestMapping(value = "/deletePerson", method = RequestMethod.GET)
public ModelAndView deletePerson(HttpServletRequest request) {
    int personId = Integer.parseInt(request.getParameter("id"));
    personDao.delete(personId);
    return new ModelAndView("redirect:/");
}

}
```

JSP (home.jsp)

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Contact Manager Home</title>
    </head>
    <body>
        <div align="center">
            <h1>Contact List</h1>
            <h3><a href="/Spring_MVC_WithJdbc_Example/newContact">New Contact</a></h3>
            <table border="1">
                <th>No</th>
                <th>Name</th>
                <th>Email</th>
                <th>Address</th>
                <th>Telephone</th>
                <th>Action</th>

                <c:forEach var="contact" items="${listContact}" varStatus="status">
                    <tr>
                        <td>${status.index + 1}</td>
                        <td>${contact.name}</td>
                        <td>${contact.email}</td>
                        <td>${contact.address}</td>
                        <td>${contact.telephone}</td>
                        <td>
                            <a href="/editContact?id=${contact.id}">Edit</a>
  

  

  

                            <a href="/deleteContact?id=${contact.id}">Delete</a>

```

```

        </td>

    </tr>
</c:forEach>
</table>
</div>
</body>
</html>

```

Person.jsp

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>New/Edit Person</title>
</head>
<body>
    <div align="center">
        <h1>New/Edit Person</h1>
        <form:form action="savePerson" method="post" modelAttribute="person">
            <table>
                <form:hidden path="id"/>
                <tr>
                    <td>Name:</td>
                    <td><form:input path="name" /></td>
                </tr>
                <tr>
                    <td>Email:</td>
                    <td><form:input path="email" /></td>
                </tr>
                <tr>
                    <td>Address:</td>
                    <td><form:input path="address" /></td>
                </tr>
                <tr>
                    <td>Telephone:</td>
                    <td><form:input path="telephone" /></td>
                </tr>
                <tr>
                    <td colspan="2" align="center"><input type="submit" value="Save"></td>
                </tr>
            </table>
        </form:form>
    </div>

```

```
    </table>
  </form:form>
</div>
</body>
</html>
```