Ads by Project Wonderful! Your ad here, right now: $0.50

Your Ads Belongs Here

Spring 3.0 MVC with Hibernate 3.0 CRUD Example

Posted by Dinesh Rajput

In this example show how to write a simple web based application with CRUD operation using Spring3 MVC Framwork with Hibernate3 using Annotation, which can handle CRUD inside its controllers. To start with it, let us have working STS IDE in place and follow the following steps to develop a Dynamic Form based Web Application using **Spring Web Framework**:

Step 1: Create a Database ***DAVDB*** on ***MySql*** Database and also we create ***Employee*** table on this database.

```
CREATE TABLE Employee(
   EMPID   INT NOT NULL AUTO_INCREMENT,
   EMPNAME VARCHAR(20) NOT NULL,
   EMPAGE  INT NOT NULL,
   SALARY BIGINT NOT NULL,
   ADDRESS VARCHAR(20) NOT NULL
   PRIMARY KEY (ID)
);
```

Step 2: Create a ***database.properties*** for database configuration information in the ***resources*** folder under ***src*** folder in the created project.

```
database.driver=com.mysql.jdbc.Driver
database.url=jdbc:mysql://localhost:3306/DAVDB
database.user=root
database.password=root
hibernate.dialect=org.hibernate.dialect.MySQLDialect
hibernate.show_sql=true
hibernate.hbm2ddl.auto=update
```

Step 3: Create a Dynamic Web Project with a name ***Spring3HibernateApp*** and create packages ***com.dineshonjava.controller***, ***com.dineshonjava.bean***, ***com.dineshonjava.dao***, ***com.dineshonjava.service***, ***com.dineshonjava.model*** under the ***src*** folder in the created project.

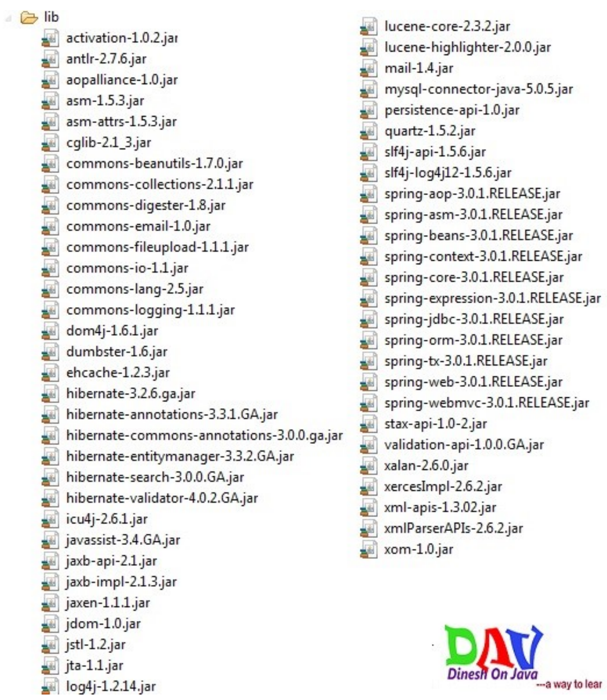
Step 4: Add below mentioned ***Spring 3.0 and Hibernate 3.0*** related libraries and other libraries into the folder ***WebRoot/WEB-INF/lib***.

TRAINING @DOJ SOFTWARE

YS JAGAN VS AP CM
Politics
Movies
New Releases
Gossips
Telugu Journalist is a widely read news portal. It comes in both English and Telugu versions. For accurate real time news Telugu Journalist stands
TELUGU JOURNALIST
CLICK HERE

LABELS

- About My Professional Life (1)
- AJAX (4)
- cloud computing (17)
- collection (17)
- Core JAVA (59)
- Garbage Collection (2)
- GitHub (3)
- Gradle (3)
- Hadoop (14)
- Hibernate (11)
- hibernate4 (4)
- interview questions (1)
- Java (2)
- JavaMail (14)
- JAXB (4)
- JDBC (1)
- JSP (29)
- JSTL (44)
- Linux (2)
- maven (3)
- migration (1)
- mongodb (18)
- motivational (1)
- multithreading (16)
- REST (18)
- Servlet (25)
- SOAP (9)
- Spring Batch (17)
- Spring Batch3 (1)
- Spring Boot (12)
- Spring3.0 (79)
- Spring4 (10)
- SpringSecurity (11)
- String in Java (31)
- struts2 (49)
- thymeleaf (1)
- UDDI (1)
- WebService (10)
- WSDL (9)



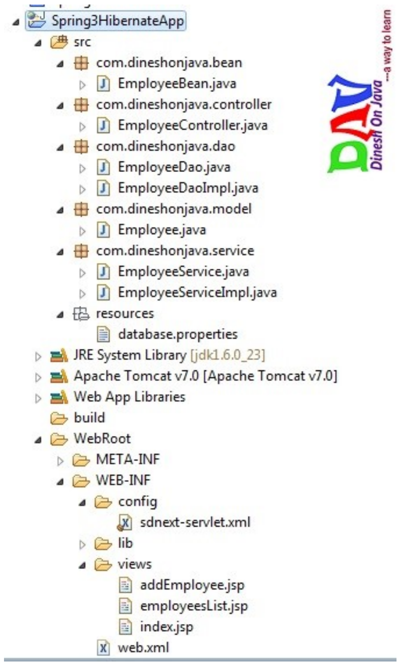
```
lib
  activation-1.0.2.jar
  antlr-2.7.6.jar
  aopalliance-1.0.jar
  asm-1.5.3.jar
  asm-attrs-1.5.3.jar
  cglib-2.1_3.jar
  commons-beanutils-1.7.0.jar
  commons-collections-2.1.1.jar
  commons-digester-1.8.jar
  commons-email-1.0.jar
  commons-fileupload-1.1.1.jar
  commons-io-1.1.jar
  commons-lang-2.5.jar
  commons-logging-1.1.1.jar
  dom4j-1.6.1.jar
  dumbster-1.6.jar
  ehcache-1.2.3.jar
  hibernate-3.2.6.ga.jar
  hibernate-annotations-3.3.1.GA.jar
  hibernate-commons-annotations-3.0.0.ga.jar
  hibernate-entitymanager-3.3.2.GA.jar
  hibernate-search-3.0.0.GA.jar
  hibernate-validator-4.0.2.GA.jar
  icu4j-2.6.1.jar
  javassist-3.4.GA.jar
  jaxb-api-2.1.jar
  jaxb-impl-2.1.3.jar
  jaxen-1.1.1.jar
  jdom-1.0.jar
  jstl-1.2.jar
  jta-1.1.jar
  log4j-1.2.14.jar
  lucene-core-2.3.2.jar
  lucene-highlighter-2.0.0.jar
  mail-1.4.jar
  mysql-connector-java-5.0.5.jar
  persistence-api-1.0.jar
  quartz-1.5.2.jar
  slf4j-api-1.5.6.jar
  slf4j-log4j12-1.5.6.jar
  spring-aop-3.0.1.RELEASE.jar
  spring-asm-3.0.1.RELEASE.jar
  spring-beans-3.0.1.RELEASE.jar
  spring-context-3.0.1.RELEASE.jar
  spring-core-3.0.1.RELEASE.jar
  spring-expression-3.0.1.RELEASE.jar
  spring-jdbc-3.0.1.RELEASE.jar
  spring-orm-3.0.1.RELEASE.jar
  spring-tx-3.0.1.RELEASE.jar
  spring-web-3.0.1.RELEASE.jar
  spring-webmvc-3.0.1.RELEASE.jar
  stax-api-1.0-2.jar
  validation-api-1.0.0.GA.jar
  xalan-2.6.0.jar
  xercesImpl-2.6.2.jar
  xml-apis-1.3.02.jar
  xmlParserAPIs-2.6.2.jar
  xom-1.0.jar
```

Step 5: Create a Java class ***EmployeeController, EmployeeBean, Employee, EmployeeDao, EmployeeDaoImpl, EmployeeService, EmployeeServiceImpl*** under the respective packages..

Step 6: Create Spring configuration files ***web.xml*** and ***sdnext-servlet.xml*** under the ***WebRoot/WEB-INF/* and *WebRoot/WEB-INF/config*** folders.

Step 7: Create a sub-folder with a name ***views*** under the ***WebRoot/WEB-INF*** folder. Create a view file ***addEmployee.jsp, employeesList.jsp and index.jsp*** under this sub-folder.

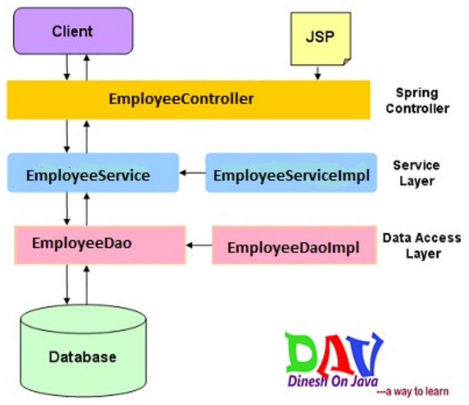
Step 8: The final step is to create the content of all the source and configuration files name ***sdnext-servlet.xml*** under the sub-folder ***/WebRoot/WEB-INF/config*** and export the application as explained below.

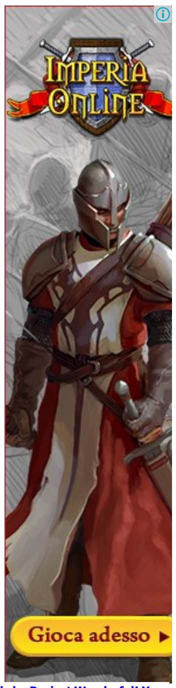


```
Spring3HibernateApp
  src
    com.dineshonjava.bean
      EmployeeBean.java
    com.dineshonjava.controller
      EmployeeController.java
    com.dineshonjava.dao
      EmployeeDao.java
      EmployeeDaoImpl.java
    com.dineshonjava.model
      Employee.java
    com.dineshonjava.service
      EmployeeService.java
      EmployeeServiceImpl.java
    resources
      database.properties
  JRE System Library [jdk1.6.0_23]
  Apache Tomcat v7.0 [Apache Tomcat v7.0]
  Web App Libraries
  build
  WebRoot
    META-INF
    WEB-INF
      config
        sdnext-servlet.xml
      lib
      views
        addEmployee.jsp
        employeesList.jsp
        index.jsp
      web.xml
```

APPLICATION ARCHITECTURE

We will have a layered architecture for our demo application. The database will be accessed by a Data Access layer popularly called as DAO Layer. This layer will use Hibernate API to interact with database. The DAO layer will be invoked by a service layer. In our application we will have a Service interface called ***EmployeeService***.





Ads by Project Wonderful! Your ad here, right now: $0.03

YS JAGAN VS AP CM
Politics
Movies
New Releases
TELUGU JOURNALIST
brings you the latest news on current happenings in the society and in and around the nation and the world.
CLICK HERE

EmployeeBean.java

```
package com.dineshonjava.bean;

/**
 * @author Dinesh Rajput
 *
 */
public class EmployeeBean {
 private Integer id;
 private String name;
 private Integer age;
 private Long salary;
 private String address;

 public Long getSalary() {
  return salary;
 }
 public void setSalary(Long salary) {
  this.salary = salary;
 }
 public Integer getId() {
  return id;
 }
 public void setId(Integer id) {
  this.id = id;
 }
 public String getName() {
  return name;
 }
 public void setName(String name) {
  this.name = name;
 }
 public Integer getAge() {
  return age;
 }
 public void setAge(Integer age) {
  this.age = age;
 }
 public String getAddress() {
  return address;
 }
 public void setAddress(String address) {
  this.address = address;
 }
}
```

Employee.java

```
package com.dineshonjava.model;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

/**
 * @author Dinesh Rajput
 *
 */
@Entity
@Table(name="Employee")
public class Employee implements Serializable{

 private static final long serialVersionUID = -723583058586873479L;

 @Id
 @GeneratedValue(strategy=GenerationType.AUTO)
 @Column(name = "empid")
 private Integer empId;

 @Column(name="empname")
 private String empName;

 @Column(name="empaddress")
 private String empAddress;

 @Column(name="salary")
 private Long salary;

 @Column(name="empAge")
 private Integer empAge;

 public Integer getEmpId() {
  return empId;
 }

 public void setEmpId(Integer empId) {
  this.empId = empId;
 }

 public String getEmpName() {
  return empName;
 }

 public void setEmpName(String empName) {
  this.empName = empName;
 }

 public String getEmpAddress() {
  return empAddress;
 }

 public void setEmpAddress(String empAddress) {
  this.empAddress = empAddress;
 }

 public Long getSalary() {
  return salary;
 }

 public void setSalary(Long salary) {
  this.salary = salary;
 }

 public Integer getEmpAge() {
  return empAge;
 }

 public void setEmpAge(Integer empAge) {
  this.empAge = empAge;
 }
}
```

EmployeeDao.java

```
package com.dineshonjava.dao;

import java.util.List;
```

```java

import com.dineshonjava.model.Employee;

/**
 * @author Dinesh Rajput
 *
 */
public interface EmployeeDao {

 public void addEmployee(Employee employee);

 public List<Employee> listEmployeess();

 public Employee getEmployee(int empid);

 public void deleteEmployee(Employee employee);
}
```

EmployeeDaoImpl.java

```java
package com.dineshonjava.dao;

import java.util.List;

import org.hibernate.SessionFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import com.dineshonjava.model.Employee;

/**
 * @author Dinesh Rajput
 *
 */
@Repository("employeeDao")
public class EmployeeDaoImpl implements EmployeeDao {

 @Autowired
 private SessionFactory sessionFactory;

 public void addEmployee(Employee employee) {
   sessionFactory.getCurrentSession().saveOrUpdate(employee);
 }

 @SuppressWarnings("unchecked")
 public List<Employee> listEmployeess() {
  return (List<Employee>)
sessionFactory.getCurrentSession().createCriteria(Employee.class).list();
 }

 public Employee getEmployee(int empid) {
  return (Employee) sessionFactory.getCurrentSession().get(Employee.class, empid);
 }

 public void deleteEmployee(Employee employee) {
  sessionFactory.getCurrentSession().createQuery("DELETE FROM Employee WHERE empid =
"+employee.getEmpId()).executeUpdate();
 }
}

```

EmployeeService.java

```java
package com.dineshonjava.service;

import java.util.List;

import com.dineshonjava.model.Employee;

/**
 * @author Dinesh Rajput
 *
 */
public interface EmployeeService {

 public void addEmployee(Employee employee);

 public List<Employee> listEmployeess();

 public Employee getEmployee(int empid);

 public void deleteEmployee(Employee employee);
}
```

EmployeeServiceImpl.java

```
package com.dineshonjava.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.dineshonjava.dao.EmployeeDao;
import com.dineshonjava.model.Employee;

/**
 * @author Dinesh Rajput
 *
 */
@Service("employeeService")
@Transactional(propagation = Propagation.SUPPORTS, readOnly = true)
public class EmployeeServiceImpl implements EmployeeService {

 @Autowired
 private EmployeeDao employeeDao;

 @Transactional(propagation = Propagation.REQUIRED, readOnly = false)
 public void addEmployee(Employee employee) {
  employeeDao.addEmployee(employee);
 }

 public List<Employee> listEmployeess() {
  return employeeDao.listEmployeess();
 }

 public Employee getEmployee(int empid) {
  return employeeDao.getEmployee(empid);
 }

 public void deleteEmployee(Employee employee) {
  employeeDao.deleteEmployee(employee);
 }

}
```

EmployeeController.java

```
package com.dineshonjava.controller;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
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.dineshonjava.bean.EmployeeBean;
import com.dineshonjava.model.Employee;
import com.dineshonjava.service.EmployeeService;

/**
 * @author Dinesh Rajput
 *
 */
@Controller
public class EmployeeController {

 @Autowired
 private EmployeeService employeeService;

@RequestMapping(value = "/save", method = RequestMethod.POST)
public ModelAndView saveEmployee(@ModelAttribute("command")EmployeeBean employeeBean,
   BindingResult result) {
  Employee employee = prepareModel(employeeBean);
  employeeService.addEmployee(employee);
  return new ModelAndView("redirect:/add.html");
 }

 @RequestMapping(value="/employees", method = RequestMethod.GET)
 public ModelAndView listEmployees() {
  Map<String Object> model = new HashMap<String Object>();
  model.put("employees",  prepareListofBean(employeeService.listEmployeess()));
  return new ModelAndView("employeesList", model);
 }

 @RequestMapping(value = "/add", method = RequestMethod.GET)
 public ModelAndView addEmployee(@ModelAttribute("command")EmployeeBean employeeBean,
   BindingResult result) {
  Map<String, Object> model = new HashMap<String, Object>();
  model.put("employees",  prepareListofBean(employeeService.listEmployeess()));
  return new ModelAndView("addEmployee", model);
 }

@RequestMapping(value = "/index", method = RequestMethod.GET)
public ModelAndView welcome() {
  return new ModelAndView("index");
 }

@RequestMapping(value = "/delete", method = RequestMethod.GET)
public ModelAndView editEmployee(@ModelAttribute("command")EmployeeBean employeeBean,
   BindingResult result) {
  employeeService.deleteEmployee(prepareModel(employeeBean));
  Map<String, Object> model = new HashMap<String, Object>();
  model.put("employee", null);
  model.put("employees",  prepareListofBean(employeeService.listEmployeess()));
  return new ModelAndView("addEmployee", model);
 }

@RequestMapping(value = "/edit", method = RequestMethod.GET)
public ModelAndView deleteEmployee(@ModelAttribute("command")EmployeeBean employeeBean,
   BindingResult result) {
  Map<String, Object> model = new HashMap<String, Object>();
  model.put("employee",
prepareEmployeeBean(employeeService.getEmployee(employeeBean.getId())));
  model.put("employees",  prepareListofBean(employeeService.listEmployeess()));
  return new ModelAndView("addEmployee", model);
 }

 private Employee prepareModel(EmployeeBean employeeBean){
  Employee employee = new Employee();
  employee.setEmpAddress(employeeBean.getAddress());
  employee.setEmpAge(employeeBean.getAge());
  employee.setEmpName(employeeBean.getName());
  employee.setSalary(employeeBean.getSalary());
  employee.setEmpId(employeeBean.getId());
  employeeBean.setId(null);
  return employee;
 }

 private List<EmployeeBean> prepareListofBean(List<Employee> employees){
  List<employeebean> beans = null;
  if(employees != null && !employees.isEmpty()){
   beans = new ArrayList<EmployeeBean>();
   EmployeeBean bean = null;
   for(Employee employee : employees){
    bean = new EmployeeBean();
    bean.setName(employee.getEmpName());
    bean.setId(employee.getEmpId());
    bean.setAddress(employee.getEmpAddress());
    bean.setSalary(employee.getSalary());
    bean.setAge(employee.getEmpAge());
    beans.add(bean);
   }
  }
  return beans;
 }

 private EmployeeBean prepareEmployeeBean(Employee employee){
  EmployeeBean bean = new EmployeeBean();
  bean.setAddress(employee.getEmpAddress());
  bean.setAge(employee.getEmpAge());
  bean.setName(employee.getEmpName());
  bean.setSalary(employee.getSalary());
  bean.setId(employee.getEmpId());
  return bean;
 }
}
```

Spring Web configuration file web.xml

```
<web-app version="2.5" 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">

   <servlet>
     <servlet-name>sdnext</servlet-name>
     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
     <init-param>
            <param-name>contextConfigLocation</param-name><param-value>/WEB-INF/config/sdnext-servlet.xml</param-value></init-param>
     <load-on-startup>1</load-on-startup>
   </servlet>

 <servlet-mapping>
  <servlet-name>sdnext</servlet-name>
  <url-pattern>*.html</url-pattern>
 </servlet-mapping>

 <welcome-file-list>
  <welcome-file>index.html</welcome-file>
 </welcome-file-list>

</web-app>
```

Spring Web configuration file sdnext-servlet.xml

```
<beans xmlns:context="http://www.springframework.org/schema/context" xmlns:tx="http://www.springframework.org/schema/tx" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.springframework.org/schema/beans" 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/tx
http://www.springframework.org/schema/tx/spring-tx-3.0.xsd">

<context:property-placeholder location="classpath:resources/database.properties">
</context:property-placeholder>
<context:component-scan base-package="com.dineshonjava">
</context:component-scan>

<tx:annotation-driven transaction-manager="hibernateTransactionManager">
</tx:annotation-driven>

<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver" id="jspViewResolver">
 <property name="viewClass" value="org.springframework.web.servlet.view.JstlView"></property>
 <property name="prefix" value="/WEB-INF/views/"></property>
 <property name="suffix" value=".jsp"></property>
</bean>

```

```
<bean class="org.springframework.jdbc.datasource.DriverManagerDataSource"
id="dataSource">
 <property name="driverClassName" value="${database.driver}"></property>
 <property name="url" value="${database.url}"></property>
 <property name="username" value="${database.user}"></property>
 <property name="password" value="${database.password}"></property>
</bean>

<bean class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean"
id="sessionFactory">
 <property name="dataSource" ref="dataSource"></property>
 <property name="annotatedClasses">
  <list>
   <value>com.dineshonjava.model.Employee</value>
  </list>
 </property>
 <property name="hibernateProperties">
 <props>
  <prop key="hibernate.dialect">${hibernate.dialect}</prop>
  <prop key="hibernate.show_sql">${hibernate.show_sql}</prop>
  <prop key="hibernate.hbm2ddl.auto">${hibernate.hbm2ddl.auto}  </prop>
        </props>
      </property>
</bean>

  <bean class="org.springframework.orm.hibernate3.HibernateTransactionManager"
id="hibernateTransactionManager">
 <property name="sessionFactory" ref="sessionFactory"></property>
  </bean>
</beans>
```

addEmployee.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!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=ISO-8859-1">
  <title>Spring MVC Form Handling</title>
 </head>
 <body>
  <h2>Add Employee Data</h2>
  <form:form method="POST" action="/sdnext/save.html">
      <table>
       <tr>
           <td><form:label path="id">Employee ID:</form:label></td>
           <td><form:input path="id" value="${employee.id}" readonly="true"/></td>
       </tr>
       <tr>
           <td><form:label path="name">Employee Name:</form:label></td>
           <td><form:input path="name" value="${employee.name}"/></td>
       </tr>
       <tr>
           <td><form:label path="age">Employee Age:</form:label></td>
           <td><form:input path="age" value="${employee.age}"/></td>
       </tr>
       <tr>
           <td><form:label path="salary">Employee Salary:</form:label></td>
           <td><form:input path="salary" value="${employee.salary}"/></td>
       </tr>

       <tr>
           <td><form:label path="address">Employee Address:</form:label></td>
                    <td><form:input path="address" value="${employee.address}"/></td>
       </tr>
          <tr>
         <td colspan="2"><input type="submit" value="Submit"/></td>
        </tr>
   </table>
  </form:form>

  <c:if test="${!empty employees}">
  <h2>List Employees</h2>
 <table align="left" border="1">
  <tr>
   <th>Employee ID</th>
   <th>Employee Name</th>
   <th>Employee Age</th>
   <th>Employee Salary</th>
   <th>Employee Address</th>
           <th>Actions on Row</th>
  </tr>

  <c:forEach items="${employees}" var="employee">
   <tr>
    <td><c:out value="${employee.id}"/></td>
    <td><c:out value="${employee.name}"/></td>
    <td><c:out value="${employee.age}"/></td>
    <td><c:out value="${employee.salary}"/></td>
    <td><c:out value="${employee.address}"/></td>
    <td align="center"><a href="edit.html?id=${employee.id}">Edit</a> | <a
href="delete.html?id=${employee.id}">Delete</a></td>
   </tr>
  </c:forEach>
 </table>
</c:if>
 </body>
</html>
```

employeesList.jsp

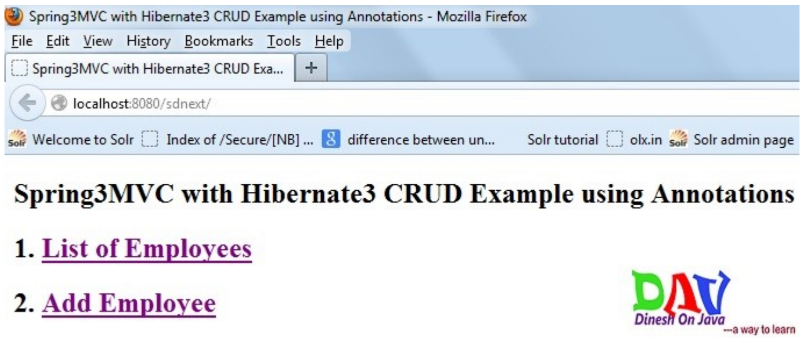
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org
/TR/html4/loose.dtd">
<html>
<head>
<title>All Employees</title>
</head>
<body>
<h1>List Employees</h1>
<h3><a href="add.html">Add More Employee</a></h3>

<c:if test="${!empty employees}">
 <table align="left" border="1">
  <tr>
   <th>Employee ID</th>
   <th>Employee Name</th>
   <th>Employee Age</th>
   <th>Employee Salary</th>
   <th>Employee Address</th>
  </tr>

  <c:forEach items="${employees}" var="employee">
   <tr>
    <td><c:out value="${employee.id}"/></td>
    <td><c:out value="${employee.name}"/></td>
    <td><c:out value="${employee.age}"/></td>
    <td><c:out value="${employee.salary}"/></td>
    <td><c:out value="${employee.address}"/></td>
   </tr>
  </c:forEach>
 </table>
</c:if>
</body>
</html>
```

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">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Spring3MVC with Hibernate3 CRUD Example using Annotations</title>
  </head>
  <body>
    <h2>Spring3MVC with Hibernate3 CRUD Example using Annotations</h2>
    <h2>1. <a href="employees.html">List of Employees</a></h2>
    <h2>2. <a href="add.html">Add Employee</a></h2>
  </body>
</html>
```

Once you are done with creating source and configuration files, export your application. Right click on your application and use **Export-> WAR** File option and save your ***Spring3HibernateApp.war*** file in Tomcat's webapps folder.

Now start your Tomcat server and make sure you are able to access other web pages from webapps folder using a standard browser. Now try a URL ***http://localhost:8080/sdnext/*** and you should see the following result if everything is fine with your Spring Web Application:

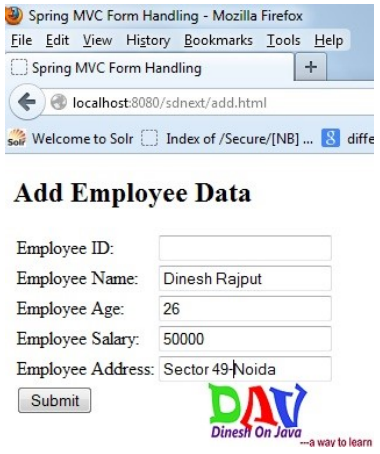

Spring3MVC with Hibernate3 CRUD Example using Annotations - Mozilla Firefox
File Edit View History Bookmarks Tools Help
localhost:8080/sdnext/
Welcome to Solr | Index of /Secure/[NB] ... | difference between un... | Solr tutorial | olx.in | Solr admin page
Dinesh On Java ---a way to learn

Spring3MVC with Hibernate3 CRUD Example using Annotations

1. List of Employees

2. Add Employee

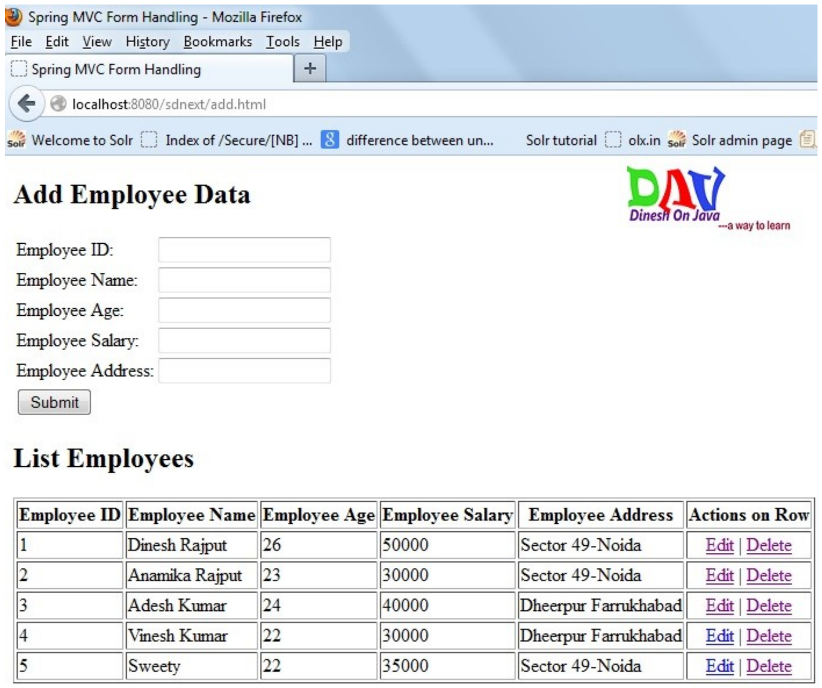
1. CREATE EMPLOYEE: Now click on the Add Employee link we get the following form for **Create Employee**.


Spring MVC Form Handling - Mozilla Firefox
File Edit View History Bookmarks Tools Help
localhost:8080/sdnext/add.html
Submit
Dinesh On Java ---a way to learn

Add Employee Data

Employee ID:	
Employee Name:	Dinesh Rajput
Employee Age:	26
Employee Salary:	50000
Employee Address:	Sector 49-Noida

Now click on the Submit button data are saved to the employee table on the database. And we get the following


Spring MVC Form Handling - Mozilla Firefox
File Edit View History Bookmarks Tools Help
localhost:8080/sdnext/add.html
Submit
Dinesh On Java ---a way to learn

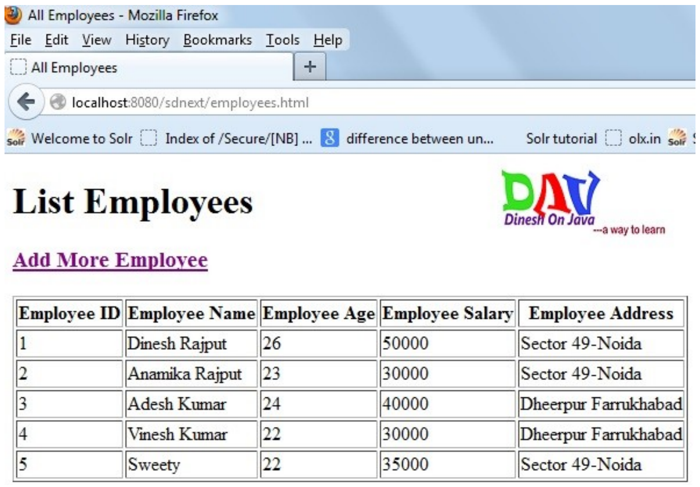
Add Employee Data

Employee ID:	
Employee Name:	
Employee Age:	
Employee Salary:	
Employee Address:	

List Employees

Employee ID	Employee Name	Employee Age	Employee Salary	Employee Address	Actions on Row
1	Dinesh Rajput	26	50000	Sector 49-Noida	Edit | Delete
2	Anamika Rajput	23	30000	Sector 49-Noida	Edit | Delete
3	Adesh Kumar	24	40000	Dheerpur Farrukhabad	Edit | Delete
4	Vinesh Kumar	22	30000	Dheerpur Farrukhabad	Edit | Delete
5	Sweety	22	35000	Sector 49-Noida	Edit | Delete

2 READ EMLOYEE : Now click on the **List of Employee** link we get the following employee list.


All Employees - Mozilla Firefox
File Edit View History Bookmarks Tools Help
All Employees
localhost:8080/sdnext/employees.html
Welcome to Solr
Index of /Secure/[NB] ...
difference between un...
Solr tutorial
olx.in
Dinesh On Java
---a way to learn

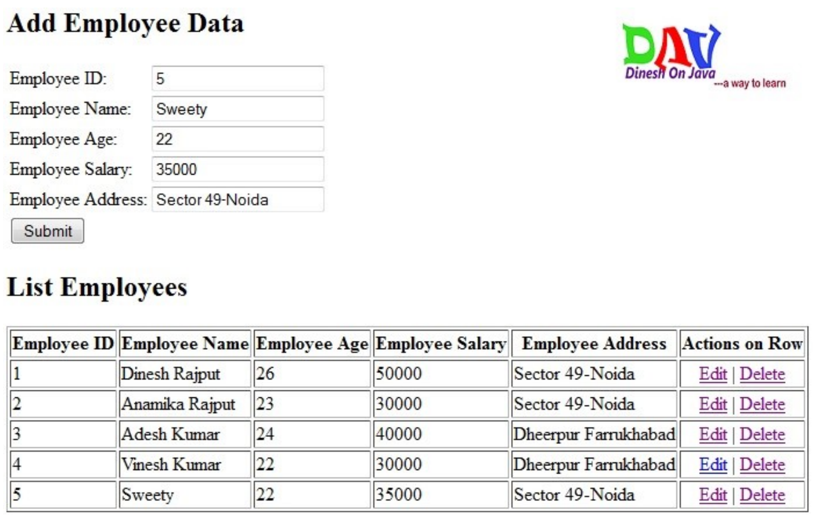
List Employees

Add More Employee

Employee ID	Employee Name	Employee Age	Employee Salary	Employee Address
1	Dinesh Rajput	26	50000	Sector 49-Noida
2	Anamika Rajput	23	30000	Sector 49-Noida
3	Adesh Kumar	24	40000	Dheerpur Farrukhabad
4	Vinesh Kumar	22	30000	Dheerpur Farrukhabad
5	Sweety	22	35000	Sector 49-Noida

3.UPDATE EMPLOYEE: for update the emloyee form we have to click on the **Edit** link in the table of employees show on the broswer.

we click Edit button for fifth record of table the >>



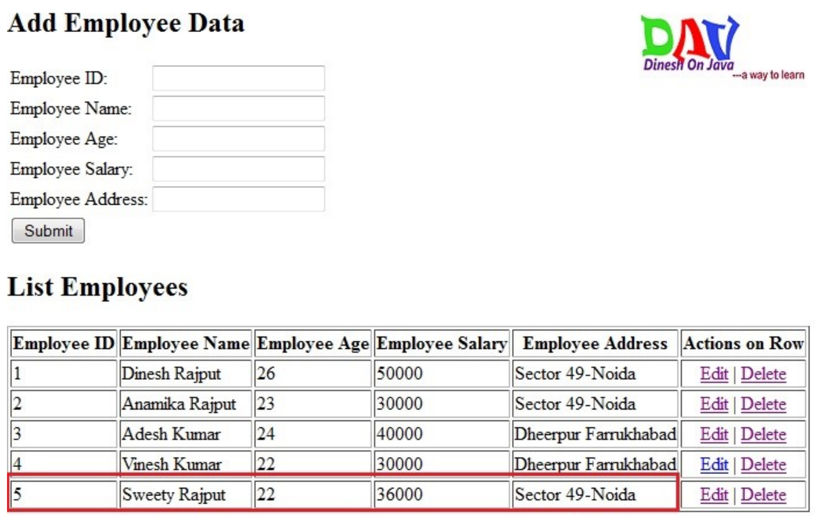
Add Employee Data

Employee ID:	5
Employee Name:	Sweety
Employee Age:	22
Employee Salary:	35000
Employee Address:	Sector 49-Noida

List Employees

Employee ID	Employee Name	Employee Age	Employee Salary	Employee Address	Actions on Row
1	Dinesh Rajput	26	50000	Sector 49-Noida	Edit | Delete
2	Anamika Rajput	23	30000	Sector 49-Noida	Edit | Delete
3	Adesh Kumar	24	40000	Dheerpur Farrukhabad	Edit | Delete
4	Vinesh Kumar	22	30000	Dheerpur Farrukhabad	Edit | Delete
5	Sweety	22	35000	Sector 49-Noida	Edit | Delete

Now update the ***name field*** value from ***Sweety to Sweety Rajput*** and ***salary filed*** value ***35000 to 36000*** and submit the data after submit we get the following updaed table against fifth row of table.



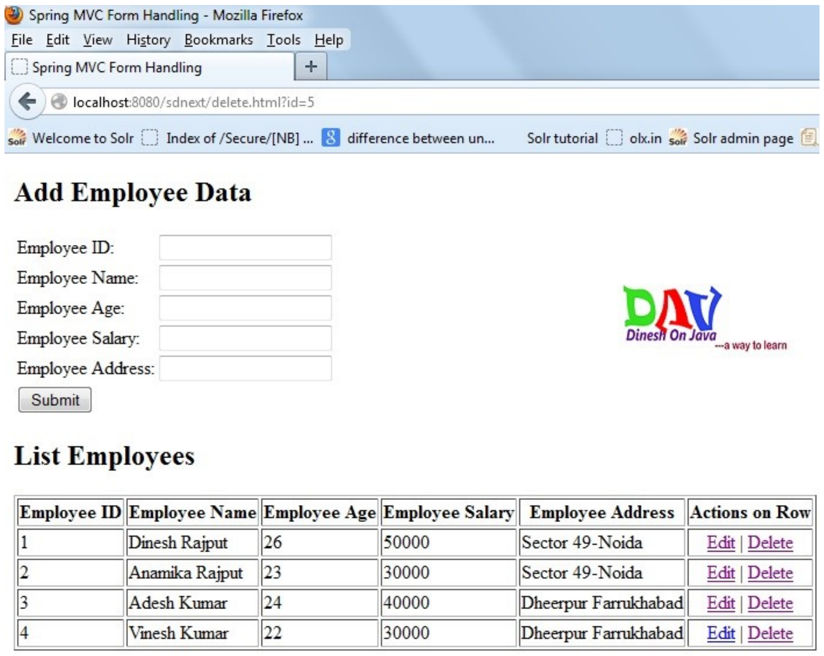
Add Employee Data

Employee ID:	
Employee Name:	
Employee Age:	
Employee Salary:	
Employee Address:	

List Employees

Employee ID	Employee Name	Employee Age	Employee Salary	Employee Address	Actions on Row
1	Dinesh Rajput	26	50000	Sector 49-Noida	Edit | Delete
2	Anamika Rajput	23	30000	Sector 49-Noida	Edit | Delete
3	Adesh Kumar	24	40000	Dheerpur Farrukhabad	Edit | Delete
4	Vinesh Kumar	22	30000	Dheerpur Farrukhabad	Edit | Delete
5	Sweety Rajput	22	36000	Sector 49-Noida	Edit | Delete

4.DELETE EMPLOYEE: Now we want to delete the one employee record from the table the we click on the delete button. we want to delete fifth records of the above table now click on the delete button of the corresponding records and we get the final list as below.


Spring MVC Form Handling - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Spring MVC Form Handling
localhost:8080/sdnext/delete.html?id=5
Welcome to Solr
Index of /Secure/[NB] ...
difference between un...
Solr tutorial
olx.in
Solr admin page
Submit
Dinesh On Java
---a way to learn

Add Employee Data

Employee ID:	
Employee Name:	
Employee Age:	
Employee Salary:	
Employee Address:	

List Employees

Employee ID	Employee Name	Employee Age	Employee Salary	Employee Address	Actions on Row
1	Dinesh Rajput	26	50000	Sector 49-Noida	Edit | Delete
2	Anamika Rajput	23	30000	Sector 49-Noida	Edit | Delete
3	Adesh Kumar	24	40000	Dheerpur Farrukhabad	Edit | Delete
4	Vinesh Kumar	22	30000	Dheerpur Farrukhabad	Edit | Delete

Download Source code
Spring3HibernateApp.zip

<<Spring Web MVC Framework |index| Spring 3.0 MVC with Tiles Example>>

6.2K

G+1 +10 Recommend this on Google

297 Comments **dineshonjava** 1 **Login**

Recommend 12 **Share** **Sort by Best**

Join the discussion…

Hema • 2 years ago
Hi Dinesh,

I have a doubt about @modelattribut("command"), actually in this what is meant by command? how it is binding with model and view? i changed the command name to emp, the add, edit operations are not working. could you pls help me out?

I look forward your reply soon,

Thanks in advance

Regards,
Hema.
24 • Reply • Share ›

Benyamine Malki Hema • a year ago
You fix it?
• Reply • Share ›

varun • 3 years ago
HTTP Status 500 - Request processing failed; nested exception is org.hibernate.exception.GenericJDBCException: could not insert: [com.dineshonjava.model.Employee]

java.sql.SQLException: Field 'ADDRESS' doesn't have a default value

PLEASE help me on this
11 • Reply • Share ›

Hien Nguyen Thanh varun • a month ago
1. you need download jar file: servlet-api.jar, log4j-1.2.17.jar

2 You need create table with mysql command:

```
CREATE TABLE IF NOT EXISTS `Employee` (
`empid` int(11) NOT NULL AUTO_INCREMENT,
`empname` varchar(20) NOT NULL,
`empAge` int(11) NOT NULL,
`salary` bigint(11) NOT NULL,
`empaddress` varchar(20) NOT NULL,
PRIMARY KEY (`empid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO_INCREMENT=3 ;
```

thanks tutorial !!!
• Reply • Share ›

Massimo varun • 2 years ago
hi Varun,
issue about java.sql.SQLException: Field 'ADDRESS' is in Entity class, move column name from empaddress to address or change column name in table
• Reply • Share ›

Prasanth • 2 years ago
Hi Dinesh, Thanks for the tutorial , I am getting the below error.
Error :- org.springframework.beans.factory.xml.XmlBeanDefinitionStoreException: Line 13 in XML document from ServletContext resource [/WEB-INF/config/psteam-servlet.xml] is invalid; nested exception is org.xml.sax.SAXParseException: cvc-elt.1: Cannot find the declaration of element 'beans'. : Line 13 in XML document from ServletContext resource [/WEB-INF/config/psteam-servlet.xml] is invalid; nested exception is org.xml.sax.SAXParseException: cvc-elt.1: Cannot find the declaration of element 'beans'.

Below is my dispatcher servlet.

```
<beans xmlns="http://www.springframework.org..." xml="" version="1.0" encoding="UTF-8" ?="">
<beans xmlns="http://www.springframework.org..." beans="" xmlns="http://www.springframework.org..."
xmlns:xsi="http://www.w3.org/2001/XMLSche..." ="http:="" www.w3.org="" 2001="" xmlschema-
instance"="" xmlns:context="http://www.springframework.org..." ="http:="" www.springframework.org=""
schema="" context"="" xmlns:tx="http://www.springframework.org..." ="http:=""
www.springframework.org="" schema="" tx"="" xmlns:p="http://www.springframework.org..." ="http:=""
www.springframework.org="" schema="" p"xsi:schemalocation="
http://www.springframework.org...
```

see more

Newer Post Home Older Post

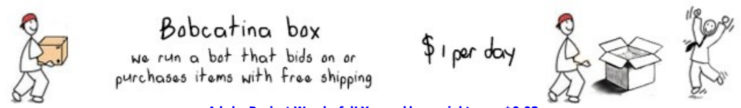
Subscribe to: Post Comments (Atom)

ADS

Ads by Project Wonderful! Your ad here, right now: $0.50

YS JAGAN VS AP CM
Politics
Movies
New Release
TELUGU JOURNALIST
CLICK HERE
Right from politics to movies including healthy gossips, Telugu Journalist stands in the forefront of real truthful journalism

Ads by Project Wonderful! Your ad here, right now: $0.03


Bobcating box
we run a bot that bids on or purchases items with free shipping
$1 per day

Ads by Project Wonderful! Your ad here, right now: $0.03

POPULAR POSTS

Spring 3.0 MVC with Hibernate 3.0 CRUD Example
In this example show how to write a simple web based application with CRUD operation using Spring3 MVC Framwork with Hibernate3 using Annot...

Core JAVA Tutorial - Core JAVA a Baby Step to be a Best Java ian
Hello Friends, Now we will focused on the Core Java tutorial, it is really a baby step to be a good, better and best Java ian :-) . I...

Spring Tutorial - Spring 3.0 a Baby Step to Learn
In this series of spring tutorials, it's provides many step by step examples and explanations on using Spring framework. The Spring framew...

Spring Security Tutorial take a Baby step to be Secure
In this spring security tutorial we will discuss about some of the security tips about the Spring Framework. Spring Security is a powerful a...

How does java Hashmap work internally
What is Hashing? Hashing in its simplest form, is a way to assigning a unique code for any variable/object after applying any formula/algor...

REST and SOAP Web Service Interview Questions
In this interview questions tutorial we will explain most asking interviews questions on the web services like SOAP, REST etc and its proto...

Spring Web MVC Framework : Chapter 38
Model view controller is a software architecture design pattern. It provides solution to layer an application by separating three concerns...

Spring Batch Tutorial - Spring Batch Process with Example : Chapter 40
Hi In this spring batch tutorial I will discuss about one of the excellent feature of Spring Framework name Spring Batch. Spring Batch is a ...

Hibernate Tutorial - Hibernate 3 on Baby Steps
This hibernate tutorial provide step by step instructions on using Hibernate 3.0. Hibernate is popular open source object rel...

Using NamedParameterJdbcTemplate in Spring with Example
The NamedParameterJdbcTemplate class helps you specify the named parameters inste a d of classic placeholder('?') argument. Named pa...

Live Traffic Feed

A visitor from Hanoi, Ha Noi viewed "Spring 3.0 MVC with Hibernate 3.0 CRUD Example | DOJ Software Consultant | Dinesh on Java" 20 secs ago

A visitor from Frisco, Texas viewed "REST and SOAP Web Service Interview Questions | DOJ Software Consultant | Dinesh on Java" 1 min ago

A visitor from Thailand viewed "Spring Simple JdbcTemplate example | DOJ Software Consultant | Dinesh on Java" 7 mins ago

A visitor from Hanoi, Ha Noi viewed "Struts 2 And JSON Example | DOJ Software Consultant | Dinesh on Java" 7 mins ago

A visitor from Sacramento, California viewed "Implementing Inheritance in Hibernate (Single Table Strategy, With Table Per Class Strategy, With Joined Strategy) : Chapter 22 | DOJ Software Consultant | Dinesh on Java" 10 mins ago

A visitor from United States viewed "Proxy Object and Eager & Lazy Fetch Types in Hibernate : Chapter 16 | DOJ Software Consultant | Dinesh on Java" 10 mins ago

A visitor from Hyderabad, Andhra Pradesh viewed "HQL Where Clause Example | DOJ Software Consultant | Dinesh on Java" 13 mins ago

A visitor from Mexico viewed "Java tutorial, Spring tutorial, Hibernate tutorial, Web Services Tutorial, Spring Batch, JAXB tutorial, Struts2 tutorial | DOJ Software Consultant | Dinesh on Java" 14 mins ago

A visitor from San Francisco, California viewed "JAX-WS Web Service Example Using Eclipse(STS) | DOJ Software Consultant | Dinesh on Java" 14 mins ago

A visitor from Ireland viewed "Struts Tutorial - Struts 2 Baby Step to Learn | DOJ Software Consultant | Dinesh on Java" 18

Real-time view · Get Feedjit