



Know why this happened →
to your page load time.



About Eren Avsarogullari



JSF 2, PrimeFaces 3, Spring 3 & Hibernate 4 Integration Project

by Eren Avsarogullari on April 9th, 2012 | Filed in: Enterprise Java Tags: JBoss Hibernate, JSF, PrimeFaces, Spring

This article shows how to integrate JSF2, PrimeFaces3, Spring3 and Hibernate4 Technologies. It provides a general project template for Java developers.

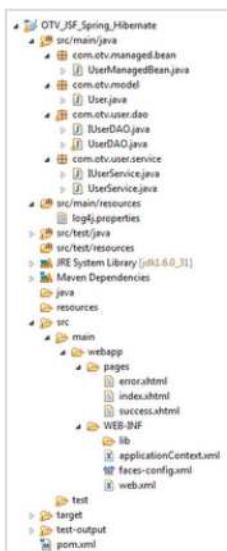
Also if Spring is not used for Business and Data Access layer, JSF – PrimeFaces & Hibernate Integration Project can be offered.

Used Technologies :

- JDK 1.6.0_31
- Spring 3.1.1
- JSF 2.1
- Hibernate 4.1.0
- Primefaces 3.1.1
- MySQL Connector 5.1.17
- MySQL 5.5.8
- c3p0 0.9.1.2
- Tomcat 7.0
- Maven 3.0.2

STEP 1 : CREATE MAVEN PROJECT

A maven project is created as below. (It can be created by using Maven or IDE Plug-in).



STEP 2 : CREATE USER TABLE

A new USER Table is created by executing below script:

```

1 CREATE TABLE USER (
2     id int(11) NOT NULL,
3     name varchar(45) NOT NULL,
4     surname varchar(45) NOT NULL,
5     PRIMARY KEY (`id`)
6 );

```

STEP 3 : LIBRARIES



It was the
#Feed action
In the
index.php
With the
strpos(): Empty delimiter
Stop guessing, w
Get Started

Carrer Opportunities

Java Developer – Novetta (FULL-TIME) April 3rd, 2013

Java Developer – HP L April 3rd, 2013

Java Developer – Wood C Junction, MD (FULL-TIM

Java Developer – FGM, IL (TIME) April 3rd, 2013

Java Developer – BlackR (FULL-TIME) April 3rd, 2013

Join Us

356,145 Mar u

Do you have a blog with ur content? Then, check out our program. You can also be a Code Geeks

Tags

Akka Android Tuto

Apache Hadoop A

Apache Tomcat Arch

Cloud Concurr

Eclipse EJB Google

Java 7 Java 8 Java EE

JAXB JBoss JBoss Hi

JSON JUnit JVM Lo

Oracle GlassFish

Performance and Scalab

Project Management RE

Security Sp

Spring MVC Spring Sec

Spring, JSF, Hibernate, Primefaces, MySQL and c3p0 dependencies are added to Maven's pom.xml.

```

01 <properties>
02   <spring.version>3.1.1.RELEASE</spring.version>
03 </properties>
04
05 <dependencies>
06
07   <!-- Spring 3 dependencies -->
08   <dependency>
09     <groupId>org.springframework</groupId>
10    <artifactId>spring-core</artifactId>
11    <version>${spring.version}</version>
12  </dependency>
13
14  <dependency>
15    <groupId>org.springframework</groupId>
16    <artifactId>spring-context</artifactId>
17    <version>${spring.version}</version>
18  </dependency>
19
20  <dependency>
21    <groupId>org.springframework</groupId>
22    <artifactId>spring-web</artifactId>
23    <version>${spring.version}</version>
24  </dependency>
25
26  <dependency>
27    <groupId>org.springframework</groupId>
28    <artifactId>spring-tx</artifactId>
29    <version>${spring.version}</version>
30  </dependency>
31
32  <dependency>
33    <groupId>org.springframework</groupId>
34    <artifactId>spring-orm</artifactId>
35    <version>${spring.version}</version>
36  </dependency>
37
38  <dependency>
39    <groupId>org.springframework</groupId>
40    <artifactId>spring-test</artifactId>
41    <version>${spring.version}</version>
42  </dependency>
43
44  <!-- JSF dependencies -->
45  <dependency>
46    <groupId>com.sun.faces</groupId>
47    <artifactId>jsf-api</artifactId>
48    <version>2.1.6</version>
49  </dependency>
50
51  <dependency>
52    <groupId>com.sun.faces</groupId>
53    <artifactId>jsf-impl</artifactId>
54    <version>2.1.6</version>
55  </dependency>
56
57  <dependency>
58    <groupId>javax.servlet</groupId>
59    <artifactId>jstl</artifactId>
60    <version>1.2</version>
61  </dependency>
62
63  <!-- Primefaces dependency -->
64  <dependency>
65    <groupId>org.primefaces</groupId>
66    <artifactId>primefaces</artifactId>
67    <version>3.1.1</version>
68  </dependency>
69
70  <!-- Hibernate dependencies -->
71  <dependency>
72    <groupId>org.hibernate</groupId>
73    <artifactId>hibernate-core</artifactId>
74    <version>4.1.0.Final</version>
75  </dependency>
76
77  <dependency>
78    <groupId>javassist</groupId>
79    <artifactId>javassist</artifactId>
80    <version>3.12.1.GA</version>
81  </dependency>
82
83  <!-- MySQL Java Connector dependency -->
84  <dependency>
85    <groupId>mysql</groupId>
86    <artifactId>mysql-connector-java</artifactId>
87    <version>5.1.17</version>
88  </dependency>
89
90  <!-- c3p0 dependency -->
91  <dependency>
92    <groupId>c3p0</groupId>
93    <artifactId>c3p0</artifactId>
94    <version>0.9.1.2</version>
95  </dependency>
96
97 </dependencies>
```

STEP 4 : CREATE USER MODEL CLASS

A new User Model Class is created.

```

01 package com.otv.model;
02
03 import javax.persistence.Column;
04 import javax.persistence.Entity;
05 import javax.persistence.Id;
06 import javax.persistence.Table;
07
08 /**
09  * 
10  * User Entity
11  * 
12  * @author onlinetechvision.com
13  * @since 25 Mar 2012
14  * @version 1.0.0
15  * 
16  */
17 @Entity
18 @Table(name="USER")
19 public class User {
20
21     private int id;
22     private String name;
23     private String surname;
24
25 /**
26  * Get User Id
27  * 
28  * @return int - User Id
29  */
30 @Id
31 @Column(name="ID", unique = true, nullable = false)
32 public int getId() {
33     return id;
34 }
35
36 /**
37  * Set User Id
38  * 
39  * @param int - User Id
40  */
41 public void setId(int id) {
42     this.id = id;
43 }
44
45 /**
46  * Get User Name
47  * 
48  * @return String - User Name
49  */
50 @Column(name="NAME", unique = true, nullable = false)
51 public String getName() {
52     return name;
53 }
54
55 /**
56  * Set User Name
57  * 
58  * @param String - User Name
59  */
60 public void setName(String name) {
61     this.name = name;
62 }
63
64 /**
65  * Get User Surname
66  * 
67  * @return String - User Surname
68  */
69 @Column(name="SURNAME", unique = true, nullable = false)
70 public String getSurname() {
71     return surname;
72 }
73
74 /**
75  * Set User Surname
76  * 
77  * @param String - User Surname
78  */
79 public void setSurname(String surname) {
80     this.surname = surname;
81 }
82
83 @Override
84 public String toString() {
85     StringBuffer strBuff = new StringBuffer();
86     strBuff.append("id : ").append(getId());
87     strBuff.append(", name : ").append(getName());
88     strBuff.append(", surname : ").append(getSurname());
89     return strBuff.toString();
90 }
91 }
```

STEP 5 : CREATE USER MANAGED BEAN CLASS

User Managed Bean is created.

```

001 package com.otv.managed.bean;
002
003 import java.io.Serializable;
```

```

004 import java.util.ArrayList;
005 import java.util.List;
006
007 import javax.faces.bean.ManagedBean;
008 import javax.faces.bean.ManagedProperty;
009 import javax.faces.bean.RequestScoped;
010
011 import org.springframework.dao.DataAccessException;
012
013 import com.otv.model.User;
014 import com.otv.user.service.IUserService;
015
016 /**
017 *
018 * User Managed Bean
019 *
020 * @author onlinetechvision.com
021 * @since 25 Mar 2012
022 * @version 1.0.0
023 *
024 */
025 @ManagedBean(name="userMB")
026 @RequestScoped
027 public class UserManagedBean implements Serializable {
028
029     private static final long serialVersionUID = 1L;
030     private static final String SUCCESS = "success";
031     private static final String ERROR   = "error";
032
033     //Spring User Service is injected...
034     @ManagedProperty(value="#{UserService}")
035     IUserService userService;
036
037     List<User> userList;
038
039     private int id;
040     private String name;
041     private String surname;
042
043     /**
044      * Add User
045      *
046      * @return String - Response Message
047      */
048     public String addUser() {
049         try {
050             User user = new User();
051             user.setId(getId());
052             user.setName(getName());
053             user.setSurname(getSurname());
054             getUserService().addUser(user);
055             return SUCCESS;
056         } catch (DataAccessException e) {
057             e.printStackTrace();
058         }
059
060         return ERROR;
061     }
062
063     /**
064      * Reset Fields
065      *
066      */
067     public void reset() {
068         this.setId(0);
069         this.setName("");
070         this.setSurname("");
071     }
072
073     /**
074      * Get User List
075      *
076      * @return List - User List
077      */
078     public List<User> getUserList() {
079         userList = new ArrayList<User>();
080         userList.addAll(getUserService().getUsers());
081         return userList;
082     }
083
084     /**
085      * Get User Service
086      *
087      * @return IUserService - User Service
088      */
089     public IUserService getUserService() {
090         return userService;
091     }
092
093     /**
094      * Set User Service
095      *
096      * @param IUserService - User Service
097      */
098     public void setUserService(IUserService userService) {
099         this.userService = userService;
100     }
101
102     /**
103      * Set User List
104      *
105      * @param List - User List
106      */

```

```

107
108     public void setUserList(List<User> userList) {
109         this.userList = userList;
110     }
111
112     /**
113      * Get User Id
114      *
115      * @return int - User Id
116      */
117     public int getId() {
118         return id;
119     }
120
121     /**
122      * Set User Id
123      *
124      * @param int - User Id
125      */
126     public void setId(int id) {
127         this.id = id;
128     }
129
130     /**
131      * Get User Name
132      *
133      * @return String - User Name
134      */
135     public String getName() {
136         return name;
137     }
138
139     /**
140      * Set User Name
141      *
142      * @param String - User Name
143      */
144     public void setName(String name) {
145         this.name = name;
146     }
147
148     /**
149      * Get User Surname
150      *
151      * @return String - User Surname
152      */
153     public String getSurname() {
154         return surname;
155     }
156
157     /**
158      * Set User Surname
159      *
160      * @param String - User Surname
161      */
162     public void setSurname(String surname) {
163         this.surname = surname;
164     }
165 }
```

STEP 6 : CREATE IUSERDAO INTERFACE

IUserDAO Interface provides methods of Data Access Layer. The data access layer manages all the logic to persist and retrieve the data from database.

```

01 package com.otv.user.dao;
02
03 import java.util.List;
04
05 import com.otv.model.User;
06
07 /**
08  *
09  * User DAO Interface
10  *
11  * @author onlinetechvision.com
12  * @since 25 Mar 2012
13  * @version 1.0.0
14  *
15  */
16 public interface IUserDAO {
17
18     /**
19      * Add User
20      *
21      * @param User user
22      */
23     public void addUser(User user);
24
25     /**
26      * Update User
27      *
28      * @param User user
29      */
30     public void updateUser(User user);
31
32     /**
33      * Delete User
34      *
35      * @param User user
36      */
```

```

37 public void deleteUser(User user);
38 /**
39 * Get User
40 *
41 * @param int User Id
42 */
43 public User getUserById(int id);
44
45 /**
46 * Get User List
47 *
48 */
49 public List<User> getUsers();
50
51 }

```

STEP 7 : CREATE UserDao CLASS

UserDAO Class is created by implementing IUserDAO Interface.

```

01 package com.otv.user.dao;
02
03 import java.util.List;
04
05 import com.otv.model.User;
06
07 import org.hibernate.SessionFactory;
08
09 /**
10 *
11 * User DAO
12 *
13 * @author onlinetechvision.com
14 * @since 25 Mar 2012
15 * @version 1.0.0
16 *
17 */
18
19 public class UserDao implements IUserDAO {
20
21     private SessionFactory sessionFactory;
22
23     /**
24     * Get Hibernate Session Factory
25     *
26     * @return SessionFactory - Hibernate Session Factory
27     */
28     public SessionFactory getSessionFactory() {
29         return sessionFactory;
30     }
31
32     /**
33     * Set Hibernate Session Factory
34     *
35     * @param SessionFactory - Hibernate Session Factory
36     */
37     public void setSessionFactory(SessionFactory sessionFactory) {
38         this.sessionFactory = sessionFactory;
39     }
40
41     /**
42     * Add User
43     *
44     * @param User user
45     */
46     @Override
47     public void addUser(User user) {
48         getSessionFactory().getCurrentSession().save(user);
49     }
50
51     /**
52     * Delete User
53     *
54     * @param User user
55     */
56     @Override
57     public void deleteUser(User user) {
58         getSessionFactory().getCurrentSession().delete(user);
59     }
60
61     /**
62     * Update User
63     *
64     * @param User user
65     */
66     @Override
67     public void updateUser(User user) {
68         getSessionFactory().getCurrentSession().update(user);
69     }
70
71     /**
72     * Get User
73     *
74     * @param int User Id
75     * @return User
76     */
77     @Override
78     public User getUserById(int id) {
79         List list = getSessionFactory().getCurrentSession()
80             .createQuery("from User where id=?")
81             .setParameter(0, id).list();
82         return (User)list.get(0);

```

```

83 }
84
85 /**
86 * Get User List
87 *
88 * @return List - User list
89 */
90 @Override
91 public List<User> getUsers() {
92 List list = getSessionFactory().getCurrentSession().createQuery("from User").list();
93 return list;
94 }
95
96 }
```

STEP 8 : CREATE IUserService INTERFACE

IUserService Interface provides methods to process the business logic.

```

01 package com.otv.user.service;
02
03 import java.util.List;
04
05 import com.otv.model.User;
06
07 /**
08 *
09 * User Service Interface
10 *
11 * @author onlinetechvision.com
12 * @since 25 Mar 2012
13 * @version 1.0.0
14 *
15 */
16 public interface IUserService {
17
18 /**
19 * Add User
20 *
21 * @param User user
22 */
23 public void addUser(User user);
24
25 /**
26 * Update User
27 *
28 * @param User user
29 */
30 public void updateUser(User user);
31
32 /**
33 * Delete User
34 *
35 * @param User user
36 */
37 public void deleteUser(User user);
38
39 /**
40 * Get User
41 *
42 * @param int UserId
43 */
44 public User getUserById(int id);
45
46 /**
47 * Get User List
48 *
49 * @return List - User list
50 */
51 public List<User> getUsers();
52 }
```

STEP 9 : CREATE UserService CLASS

UserService Class is created by implementing IUserService Interface.

```

01 package com.otv.user.service;
02
03 import java.util.List;
04
05 import org.springframework.transaction.annotation.Transactional;
06
07 import com.otv.model.User;
08 import com.otv.user.dao.IUserDAO;
09
10 /**
11 *
12 * User Service
13 *
14 * @author onlinetechvision.com
15 * @since 25 Mar 2012
16 * @version 1.0.0
17 *
18 */
19 @Transactional(readOnly = true)
20 public class UserService implements IUserService {
21
22 // UserDao is injected...
23 IUserDAO userDao;
```

```

24 /**
25  * Add User
26  *
27  * @param User user
28  */
29 @Transactional(readOnly = false)
30 @Override
31 public void addUser(User user) {
32     getUserDAO().addUser(user);
33 }
34
35 /**
36  * Delete User
37  *
38  * @param User user
39  */
40 @Transactional(readOnly = false)
41 @Override
42 public void deleteUser(User user) {
43     getUserDAO().deleteUser(user);
44 }
45
46 /**
47  * Update User
48  *
49  * @param User user
50  */
51 @Transactional(readOnly = false)
52 @Override
53 public void updateUser(User user) {
54     getUserDAO().updateUser(user);
55 }
56
57 /**
58  * Get User
59  *
60  * @param int User Id
61  */
62 @Override
63 public User getUserId(int id) {
64     return getUserDAO().getUserId(id);
65 }
66
67 /**
68  * Get User List
69  *
70  */
71 @Override
72 public List<User> getUsers() {
73     return getUserDAO().getUsers();
74 }
75
76 /**
77  * Get User DAO
78  *
79  * @return IUserDAO - User DAO
80  */
81 public IUserDAO getUserDAO() {
82     return userDAO;
83 }
84
85 /**
86  * Set User DAO
87  *
88  * @param IUserDAO - User DAO
89  */
90 public void setUserDAO(IUserDAO userDAO) {
91     this.userDAO = userDAO;
92 }
93 }
94 }
```

STEP 10 : CREATE applicationContext.xml

Spring Application Context's content is shown as follows :

```

01 <beans xmlns="http://www.springframework.org/schema/beans"
02   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
03   xmlns:tx="http://www.springframework.org/schema/tx"
04   xmlns:context="http://www.springframework.org/schema/context"
05   xsi:schemaLocation="http://www.springframework.org/schema/beans
06
07 http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
08
09
10 http://www.springframework.org/schema/tx
11
12
13 http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
14
15
16 http://www.springframework.org/schema/context
17
18 http://www.springframework.org/schema/context/spring-context-3.0.xsd">
19
20 <!-- Beans Declaration -->
21 <bean id="User" class="com.otv.model.User"/>
22
23 <!-- User Service Declaration -->
24 <bean id="UserService" class="com.otv.user.service.UserService">
25   <property name="userDAO" ref="UserDAO" />
26 </bean>
```

```

27
28 <!-- User DAO Declaration -->
29 <bean id="UserDAO" class="com.otv.user.dao.UserDAO">
30   <property name="sessionFactory" ref="SessionFactory" />
31 </bean>
32
33 <!-- Data Source Declaration -->
34 <bean id="DataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-method="close">
35   <property name="driverClass" value="com.mysql.jdbc.Driver" />
36   <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/Test" />
37   <property name="user" value="root" />
38   <property name="password" value="root" />
39
40   <property name="maxPoolSize" value="10" />
41   <property name="maxStatements" value="0" />
42   <property name="minPoolSize" value="5" />
43 </bean>
44
45 <!-- Session Factory Declaration -->
46 <bean id="SessionFactory" class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">
47   <property name="dataSource" ref="DataSource" />
48   <property name="annotatedClasses">
49     <list>
50       <value>com.otv.model.User</value>
51     </list>
52   </property>
53   <property name="hibernateProperties">
54     <props>
55       <prop key="hibernate.dialect">org.hibernate.dialect.MySQLDialect</prop>
56       <prop key="hibernate.show_sql">true</prop>
57     </props>
58   </property>
59 </bean>
60
61 <!-- Enable the configuration of transactional behavior based on annotations -->
62   <tx:annotation-driven transaction-manager="txManager"/>
63
64 <!-- Transaction Manager is defined -->
65   <bean id="txManager" class="org.springframework.orm.hibernate4.HibernateTransactionManager">
66     <property name="sessionFactory" ref="SessionFactory"/>
67   </bean>
68 </beans>

```

STEP 11 : CREATE faces-config.xml

JSF Configuration is shown as follows :

```

01 <?xml version="1.0" encoding="UTF-8"?>
02 <faces-config
03   xmlns="http://java.sun.com/xml/ns/javaee"
04   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
05   xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
06
07 http://java.sun.com/xml/ns/javaee/web-facesconfig_2_0.xsd"
08
09   version="2.0">
10
11 <!-- JSF and Spring are integrated -->
12 <application>
13   <el-resolver>
14     org.springframework.web.jsf.el.SpringBeanFacesELResolver
15   </el-resolver>
16 </application>
17
18 <!-- configuration of navigation rules -->
19 <navigation-rule>
20   <from-view-id>/pages/index.xhtml</from-view-id>
21   <navigation-case>
22     <from-outcome>success</from-outcome>
23     <to-view-id>/pages/success.xhtml</to-view-id>
24   </navigation-case>
25   <navigation-case>
26     <from-outcome>error</from-outcome>
27     <to-view-id>/pages/error.xhtml</to-view-id>
28   </navigation-case>
29
30 </navigation-rule>
31 </faces-config>

```

STEP 12 : CREATE web.xml

web.xml is configured as follows :

```

01 <?xml version="1.0" encoding="UTF-8"?>
02 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
03   xmlns="http://java.sun.com/xml/ns/javaee"
04   xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
05   xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
06 app_2_5.xsd"
07   id="WebApp_ID"
08   version="2.5">
09
10   <display-name>OTV_JSF_PrimeFaces_Spring_Hibernate</display-name>
11
12   <!-- Spring Context Configuration's Path definition -->
13   <context-param>
14     <param-name>contextConfigLocation</param-name>
15     <param-value>
16       /WEB-INF/applicationContext.xml

```

```

16   </param-value>
17 </context-param>
18
19   <!-- The Bootstrap listener to start up and shut down Spring's root WebApplicationContext. It is
20 registered to Servlet Container -->
21   <listener>
22     <listener-class>
23       org.springframework.web.context.ContextLoaderListener
24     </listener-class>
25   </listener>
26   <listener>
27     <listener-class>
28       org.springframework.web.context.request.RequestContextListener
29     </listener-class>
30   </listener>
31
32   <!-- Project Stage Level -->
33   <context-param>
34     <param-name>javax.faces.PROJECT_STAGE</param-name>
35     <param-value>Development</param-value>
36   </context-param>
37
38   <!-- Welcome Page -->
39   <welcome-file-list>
40     <welcome-file>/pages/index.xhtml</welcome-file>
41   </welcome-file-list>
42
43   <!-- JSF Servlet is defined to container -->
44   <servlet>
45     <servlet-name>Faces Servlet</servlet-name>
46     <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
47     <load-on-startup>1</load-on-startup>
48   </servlet>
49
50   <!-- Mapping with servlet and url for the http requests. -->
51   <servlet-mapping>
52     <servlet-name>Faces Servlet</servlet-name>
53     <url-pattern>*.jsf</url-pattern>
54   </servlet-mapping>
55
56   <servlet-mapping>
57     <servlet-name>Faces Servlet</servlet-name>
58     <url-pattern>*.faces</url-pattern>
59   </servlet-mapping>
60
61 </web-app>

```

STEP 13 : CREATE index.xhtml

index.xhtml is created as follows :

Id :	<input type="text" value="12345"/>
Name :	<input type="text" value="Bruce"/>
Surname :	<input type="text" value="Willis"/>
<input type="button" value="Add"/> <input type="button" value="Reset"/>	

```

01 <html xmlns="http://www.w3.org/1999/xhtml"
02   xmlns:h="http://java.sun.com/jsf/html"
03   xmlns:f="http://java.sun.com/jsf/core"
04   xmlns:p="http://primefaces.org/ui">
05 <h:head><title>Welcome to OTV_JSF_Spring_Hibernate_Project</title></h:head>
06 <h:body>
07   <h:form>
08     <table>
09       <tr>
10         <td><h:outputLabel for="id" value="Id : " /></td>
11         <td><p:inputText id="id" value="#{userMB.id}">
12           <f:converter converterId="javax.faces.Integer"/>
13           <p:ajax event="blur" update="idMsg" />
14         </p:inputText>
15         <p:message id="idMsg" for="id" display="icon"/>
16       </td>
17     </tr>
18     <tr>
19       <td><h:outputLabel for="name" value="Name : " /></td>
20       <td><p:inputText id="name" value="#{userMB.name}">
21         <f:validateLength minimum="5" />
22         <p:ajax event="blur" update="nameMsg" />
23       </p:inputText>
24       <p:message id="nameMsg" for="name" display="icon"/>
25     </td>
26   </tr>
27   <tr>
28     <td><h:outputLabel for="surname" value="Surname : " /></td>
29     <td><p:inputText id="surname" value="#{userMB.surname}">
30       <f:validateLength minimum="5" />
31       <p:ajax event="blur" update="surnameMsg" />
32     </p:inputText>
33     <p:message id="surnameMsg" for="surname" display="icon"/>
34   </td>
35   </tr>
36     <tr>
37       <td><p:commandButton id="addUser" value="Add" action="#{userMB.addUser}" ajax="false"/>
38       <td><p:commandButton id="reset" value="Reset" action="#{userMB.reset}" ajax="false"/></td>

```

```

39      </tr>
40    </table>
41  </h:form>
42 </h:body>
43 </html>
```

STEP 14 : CREATE success.xhtml

success.xhtml is created as follows :

USERS :		
ID	Name	Surname
12345	Bruce	Willis

```

01 <html xmlns="http://www.w3.org/1999/xhtml"
02   xmlns:h="http://java.sun.com/jsf/html"
03   xmlns:f="http://java.sun.com/jsf/core"
04   xmlns:p="http://primefaces.org/ui">
05 <h:head>
06   <title>Welcome to OTV_JSF_Spring_Hibernate_Project</title>
07 </h:head>
08 <h:body>
09   <h:form>
10     <h:outputText value="USERS : "></h:outputText>
11     <p: dataTable id="users" var="user" value="#{userMB.userList}" style="width: 10%">
12       <p:column>
13         <f:facet name="header">
14           <h:outputText value="ID" />
15         </f:facet>
16           <h:outputText value="#{user.id}" />
17       </p:column>
18       <f:facet name="header">
19         <h:outputText value="Name" />
20       </f:facet>
21           <h:outputText value="#{user.name}" />
22       </p:column>
23       <f:facet name="header">
24         <h:outputText value="Surname" />
25       </f:facet>
26           <h:outputText value="#{user.surname}" />
27       </p:column>
28     </p: dataTable>
29   </h:form>
30 </h:body>
31 </html>
```

STEP 15 : CREATE error.xhtml

error.xhtml is created as follows :

```

01 <html xmlns="http://www.w3.org/1999/xhtml"
02   xmlns:h="http://java.sun.com/jsf/html"
03   xmlns:f="http://java.sun.com/jsf/core"
04   xmlns:p="http://primefaces.org/ui">
05
06 <h:head><title>Welcome to JSF_PrimeFaces_Spring_Hibernate Project</title></h:head>
07 <body>
08   <f:view>
09     <h:form>
10       <h:outputText value="Transaction Error has occurred!"></h:outputText>
11     </h:form>
12   </f:view>
13 </body>
14 </html>
```

STEP 16 : DEPLOY PROJECT

After **OTV_JSF_Spring_Hibernate** Project is deployed to Tomcat, index page can be opened via following URL :

http://ip:port/OTV_JSF_Spring_Hibernate_Project-1.0-SNAPSHOT/pages/index.jsf

STEP 17 : DOWNLOAD

[OTV_JSF_Spring_Hibernate](#)

Reference: JSF2 + Primefaces3 + Spring3 & Hibernate4 Integration Project from our JCG partner Eren Avsarogullari at the [Online Technology Vision](#) blog.

You might also like:

- [JSF – PrimeFaces & Hibernate Integration Project](#)
- [How cool is integration testing with Spring and Hibernate](#)
- [RESTEasy Tutorial Part-2: Spring Integration](#)
- [JBoss 4.2.x Spring 3 JPA Hibernate Tutorial Part #2](#)
- [Full Web Application Tomcat JSF Primefaces JPA Hibernate – Part 3](#)

Share and enjoy!



21 comments

★ 5



Leave a message...

Best ▾ Community

Share ▾

**James Ngumba** · 11 months ago

How can one use the user DAO classes above as generic classes so that there is no repetition for each object in a project?

10 ▾ | ▾ · Reply · Share ›

Alwyn Schoeman · 11 months ago

Is there a way to specify the JSF navigation rules with spring rather than in the faces-config file?

3 ▾ | ▾ · Reply · Share ›

simont · a month ago

I don't see the benefit of using Spring since hibernate offer the persistance and jsf-primefaces offer the rich UI components with a big facility and also offer mvc architecture , what do you think ?

0 ▾ | ▾ · Reply · Share ›

Arturo Agudelo · 2 months ago

Nice

0 ▾ | ▾ · Reply · Share ›

ponic · 4 months ago

Nice simple tutorial. Great.

0 ▾ | ▾ · Reply · Share ›

ponic · 4 months ago

I would like to know which maven archetype can be used for creating the project structure for these kind of projects. E.g in the screen shot of your project as per the screen shot which is posted above, have you used any archetype? If so what is the name of it? Thanks

0 ▾ | ▾ · Reply · Share ›

Fredrik · 6 months ago

I had issues with my build path. This solved my issues.

<http://stackoverflow.com/question...>

0 ▾ | ▾ · Reply · Share ›

Pablo · 7 months ago

How do you create a maven project with that file structure? (sorry for the dumb question)

0 ▾ | ▾ · Reply · Share ›

Imad Oppa · 7 months ago

unfortunately it doesn't work for me with this config .. but when I replace Hibernate4 with Hibernate3 now it works, because you used the propriety annotatedClasses here so we need to use the class AnnotationSessionFactoryBean in SessionFactory and not LocalSessionFactoryBean and since it doesn't exist then it must go through "org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean" and use class "org.springframework.orm.hibernate3.HibernateTransactionManager" in Transaction Manager Bean

0 ▾ | ▾ · Reply · Share ›

Ganesh Kumar Kumaraswamy · 9 months ago

Tried using annotation based (package-scan) implementation, having problem injecting managed bean with Richfaces JSF. I'm not getting any error. However my bean is either not called or not visible in Face Container. Any problem which you might have faced already?

0 ▾ | ▾ · Reply · Share ›

Amine El idrissi · 9 months ago

i get the following when i start the project using tomcat:

"FAIL - Application for context path / OTV_JSF_Spring_Hibernate-1.0-SNAPSHOT could not be started"

i have to notice that after deploying the project in tomcat i "functional" is set to "false" is that has an effect.

0 ▾ | ▾ · Reply · Share ›

Ovidiu Drumia · 10 months ago

Hi,

Nice tutorial. You might want to update it with the Spring 3 specification(annotations, package-scan). Also you can

include the jetty plugin in maven to run it with the jetty:run goal.

Anyway it seems u forgot to put the:

```
<servlet-mapping>
  <servlet-name>Faces Servlet</servlet-name>
  <url-pattern>*.xhtml</url-pattern>
</servlet-mapping>
```

in the web.xml.

Best of luck,

Ovidiu

0 ▲ | ▼ · Reply · Share ›

bbkumscap · a year ago

Hi

I solved my probleme.first, it's not necessary to add some dependancies.2nd, in postgres, User is a reserved keyword , we should not use this as a table name.Thanks

0 ▲ | ▼ · Reply · Share ›

Ghazi Hakim · a year ago

Hi

I have a warning in faces-config.xml in this tag:

```
<application>
  <el-resolver>
    org.springframework.web.jsf.el.SpringBeanFacesELResolver
  </el-resolver>
</application> the warning is:Class org.springframework.web.jsf.el.SpringBeanFacesELResolver must extend the
type javax.el.ELResolver. Any solution ?
```

Thank you

0 ▲ | ▼ · Reply · Share ›

bbkumscap ➔ Ghazi Hakim · a year ago

add this in your pom

```
<dependency>
  <groupId>org.apache.tomcat</groupId>
  <artifactId>tomcat-el-api</artifactId>
  <version>${tomcat.version}</version>
  <scope>provided</scope>
</dependency>
```

Or

```
<dependency>
  <groupId>javax.el</groupId>
  <artifactId>el-api</artifactId>
  <version>2.2</version>
  <scope>provided</scope>
</dependency>
```

0 ▲ | ▼ · Reply · Share ›

bbkumscap · a year ago

hi

i download and test this sample, i little changed it for using postgres instead mysql, when i compiled it, it made errors, i resolved the problems by adding dependancies

```
<dependency> <groupId>dom4j</groupId> <artifactId>dom4j</artifactId> <version>1.6.1</version>
<scope>compile</scope> </dependency> <dependency> <groupId>org.jboss.logging</groupId> <artifactId>jboss-
logging</artifactId> <version>3.1.0.GA</version> </dependency> <dependency>
<groupId>org.hibernate</groupId> <artifactId>hibernate-infinispan</artifactId> <version>4.1.1.Final</version>
</dependency> <dependency> <groupId>org.infinispan</groupId> <artifactId>infinispan-core</artifactId>
<version>5.1.2.CR1</version> </dependency> <dependency>
<groupId>org.hibernate.javax.persistence</groupId> <artifactId>hibernate-jpa-2.0-api</artifactId>
<version>1.0.1.Final</version> </dependency> <dependency> <groupId>javax.transaction</groupId>
<artifactId>jta</artifactId> <version>1.1</version> <type>jar</type> </dependency>
```

When i add the new user, i have a transaction error

Caused by: org.postgresql.util.PSQLException: ERREUR: erreur de syntaxe sur ou près de « user » Position : 13 at
org.postgresql.core.v3.QueryExecutorImpl.receiveErrorResponse(QueryExecutorImpl.java:2103) at
org.postgresql.core.v3.QueryExecutorImpl.processResults(QueryExecutorImpl.java:1836) at
~~org.postgresql.core.v3.QueryExecutorImpl.execute(QueryExecutorImpl.java:257) at~~

see more

0 ▲ | ▼ · Reply · Share ›

bbkumscap ➔ bdkumscap · a year ago

solved

0 ▲ | ▼ · Reply · Share ›

Ghazi Hakim · a year ago

I want to use "run on server" option in Eclipse instead of deploying the war file manually in Tomcat Manager,

How can I do that ?

Thank you

0 ▲ | ▼ · Reply · Share ›

Byron Kiourtzoglou Mod · a year ago

The download link is fixed now ... sorry for the inconvenience

0 ▲ | ▼ · Reply · Share ›

Ghazi Hakim → Byron Kiourtzoglou · a year ago

Thank you :)

0 ▲ | ▼ · Reply · Share ›

Ghazi Hakim · a year ago

thank you!

but I can't download it!!

0 ▲ | ▼ · Reply · Share ›

ALSO ON JAVA CODE GEEKS

[How to Use PropertyNamingStrategy in Jackson](#)

2 comments

[Java StringBuilder myth debunked](#) 13 comments

[Function interface – A functional interface in the java.util.function package in Java 8](#) 4 comments

[Java 8 Lambdas – The missing link to moving away from Java](#) 2 comments

AROUND THE WEB

[Careful with Online Photos: Geotags Tell Your Boss Where You've Been!](#) Citi Women & Co.

[Top 10 Illustrators Online Portfolios Worth Exploring](#) My Life Scoop

[Nasal Congestion Remedies For Cold, Allergy And Sinus Relief](#) SymptomFind

[The Smartest Financial Advisor Ever? Marty McFly](#) First To Know

What's this?

[Comment feed](#)

[Subscribe via email](#)

Knowledge Base

Partners

Hall Of Fame

About Java Code Geeks

[Examples](#)

[Mkyong](#)

["Android Full Application Tutorial" series](#)

JCGs (Java Code Geeks) is an independent online community creating the ultimate Java to Java developers resource for technical architect, technical team lead (senior developer), junior developers alike. JCGs serve the Java, SOA, Agile communities with daily news written by domain experts, announcements, code snippets and open source projects.

[Resources](#)

[The Code Geeks Network](#)

[GWT 2 Spring 3 JPA 2 Hibernate 3.5 Tutorial](#)

License

[Software](#)

[Java Code Geeks](#)

[Android Game Development Tutorials](#)



Java Code Geeks content is offered under the

[Attribution-ShareAlike 3.0 Unported License](#)

distribution, you must make clear to others the license terms. The best way to do this is with a link to this web page. Any of this content can be waived if you get written permission from Java Code Geeks. This license impairs or restricts the author's moral rights.

[Tutorials](#)

[.NET Code Geeks](#)

[Android Google Maps Tutorial](#)

Comments

[Web Code Geeks](#)

[Android Location Based Services Application – GPS location](#)

Comments

[Funny Source Code Comments](#)

Comments

[Java Best Practices – Vector vs ArrayList vs HashSet](#)

Comments

[Android JSON Parsing with Gson Tutorial](#)

Comments

[Android Quick Preferences Tutorial](#)

© 2010-2012 Java Code Geeks. Licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License.

All trademarks and registered trademarks appearing on Java Code Geeks are the property of their respective owners.

Java is a trademark or registered trademark of Oracle Corporation in the United States and other countries.

Java Code Geeks is not connected to Oracle Corporation and is not sponsored by Oracle Corporation.