« Part 3 – Basic Implementation of CRUD
Part 5 – Downloading PrimeFaces themes / creating and installing custom themes and overriding PrimeFaces style sheets. »

Part 4 – Lazy Loading and Pagination with JSF 2.1 / Primefaces 3.4.2 DataTable and Form validation 28

21 Sep 2012 | Java EE Tags: JSF · Lazy Loading · PrimeFaces

Part 4 – Lazy Loading and Pagination with JSF 2.1 / Primefaces 3.5 DataTable

Integrating PrimeFaces, JSF and Maven is pretty easy. All we need to do is to add the PrimeFaces repository definition to our pom.xml file between repositories tag and then add PrimeFaces artifact in between dependencies tags as it is shown below.

After we add repository and dependency configuration to to pom.xml file. (Please download the source code to see the original pom.xml file)

Add this repository in between repositories tags.

```
<repository>
        <id>prime-repo</id>
        <name>PrimeFaces Maven Repository</name>
        <url>http://repository.primefaces.org</url>
        <layout>default</layout>
</repository>
```

Add this dependency configuration in between dependencies tags.

```
<dependency>
            <groupId>org.primefaces</groupId>
            <artifactId>primefaces</artifactId>
            <version>3.4.2</version>
  </dependency>
```

Our LazyUserDataModel class has to extend LazyDataModel so we can lazy load user list:

```
public class LazyUserDataModel extends LazyDataModel<User> implemen

    private List<User> datasource;
    private int pageSize;
```

Blogroll

› Arquillian testing tool for Java EE
› PrimeFaces JSF Component Suite

Recent Posts

› Glassfish4 and JSF 2.2 ViewScope support (JSF 2.2, CDI, JPA 2 (ORM) / EclipseLink, JAAS , MySQL, example CRUD web application)
› My Book Review – Instant PrimeFaces Starter- Packt Publishing
› Reviewing Instant PrimeFaces Starter- Packt Publishing
› My Book Review – PrimeFaces Cookbook by Oleg Varaksin and Mert Çalışkan
› Reviewing PrimeFaces Cookbook – Packt Publishing – Oleg Varaksin, Mert Çalışkan

Recent Comments

› simtay on Part 4 – Lazy Loading and Pagination with JSF 2.1 / Primefaces 3.4.2 DataTable and Form validation
› simtay on Simple CRUD Web Application with JSF 2.1, PrimeFaces 3.5, Maven and JPA
› Mohamed on Part 4 – Lazy Loading and Pagination with JSF 2.1 / Primefaces 3.4.2 DataTable and Form validation
› dukgo on Simple CRUD Web Application with JSF 2.1, PrimeFaces 3.5, Maven and JPA
› simtay on Simple CRUD Web Application with JSF 2.1, PrimeFaces 3.5, Maven and JPA

Archives

› November 2013 › August 2013
› April 2013 › March 2013
› January 2013 › September 2012

Categories

```
    private int rowIndex;
    private int rowCount;
    private DataAccessService crudService;

    public LazyUserDataModel(DataAccessService crudService) {
        this.crudService = crudService;
    }

    @Override
    public List<User> load(int first, int pageSize, String sortField
        datasource = crudService.findWithNamedQuery(User.ALL, first,
        setRowCount(crudService.countTotalRecord(User.TOTAL));
        return datasource;
    }

    @Override
    public boolean isRowAvailable() {
        if(datasource == null)
            return false;
        int index = rowIndex % pageSize ;
        return index >= 0 && index < datasource.size();
    }

    @Override
    public Object getRowKey(User user) {
        return user.getId().toString();
    }

    @Override
    public User getRowData() {
        if(datasource == null)
            return null;
        int index =  rowIndex % pageSize;
        if(index > datasource.size()){
            return null;
        }
        return datasource.get(index);
    }

    @Override
    public User getRowData(String rowKey) {
        if(datasource == null)
            return null;
       for(User user : datasource) {
           if(user.getId().toString().equals(rowKey))
           return user;
       }
       return null;
    }

    @Override
    public void setPageSize(int pageSize) {
        this.pageSize = pageSize;
    }

    @Override
    public int getPageSize() {
        return pageSize;
    }

    @Override
    public int getRowIndex() {
        return this.rowIndex;
    }

    @Override
    public void setRowIndex(int rowIndex) {
        this.rowIndex = rowIndex;
    }

    @Override
    public void setRowCount(int rowCount) {
        this.rowCount = rowCount;
    }

    @Override
    public int getRowCount() {
        return this.rowCount;
    }

    @Override
    public void setWrappedData(Object list) {
        this.datasource = (List<User>) list;
    }

    @Override
    public Object getWrappedData() {
        return datasource;
    }
```

› Java EE

Meta

› Log in
› Entries RSS
› Comments RSS
› WordPress.org

```
94  }
```

Generic DataAccessService class supporting lazy loading:

```
1   /**
2    * Implementation of the generic Data Access Service
3    * All CRUD (create, read, update, delete) basic data access operat
4    * persistent object are performed in this class.
5    * @author Emre Simtay <emre@simtay.com>
6    */
7
8   public abstract class DataAccessService<T> {
9
10      @PersistenceContext
11      private EntityManager em;
12
13      public DataAccessService() {
14      }
15
16      private Class<T> type;
17
18      /**
19       * Default constructor
20       *
21       * @param type entity class
22       */
23      public DataAccessService(Class<T> type) {
24          this.type = type;
25      }
26
27      /**
28       * Stores an instance of the entity class in the database
29       * @param T Object
30       * @return
31       */
32      public T create(T t) {
33          this.em.persist(t);
34          this.em.flush();
35          this.em.refresh(t);
36          return t;
37      }
38
39      /**
40       * Retrieves an entity instance that was previously persisted t
41       * @param T Object
42       * @param id
43       * @return
44       */
45      public T find(Object id) {
46          return this.em.find(this.type, id);
47      }
48
49      /**
50       * Removes the record that is associated with the entity instan
51       * @param type
52       * @param id
53       */
54      public void delete(Object id) {
55          Object ref = this.em.getReference(this.type, id);
56          this.em.remove(ref);
57      }
58
59      /**
60       * Removes the number of entries from a table
61       * @param <T>
62       * @param items
63       * @return
64       */
65      public boolean deleteItems(T[] items) {
66          for (T item : items) {
67              em.remove(em.merge(item));
68          }
69          return true;
70      }
71
72      /**
73       * Updates the entity instance
74       * @param <T>
75       * @param t
76       * @return the object that is updated
77       */
78      public T update(T t) {
79          return (T) this.em.merge(t);
80      }
81
82      /**
83       * Returns the number of records that meet the criteria
84       * @param namedQueryName
```

```
85          * @return List
86          */
87         public List findWithNamedQuery(String namedQueryName) {
88             return this.em.createNamedQuery(namedQueryName).getResultLi
89         }
90 
91         /**
92          * Returns the number of records that meet the criteria
93          * @param namedQueryName
94          * @param parameters
95          * @return List
96          */
97         public List findWithNamedQuery(String namedQueryName, Map param
98             return findWithNamedQuery(namedQueryName, parameters, 0);
99         }
100 
101        /**
102         * Returns the number of records with result limit
103         * @param queryName
104         * @param resultLimit
105         * @return List
106         */
107        public List findWithNamedQuery(String queryName, int resultLimi
108            return this.em.createNamedQuery(queryName).
109                    setMaxResults(resultLimit).
110                    getResultList();
111        }
112 
113        /**
114         * Returns the number of records that meet the criteria
115         * @param <T>
116         * @param sql
117         * @param type
118         * @return List
119         */
120        public List<T> findByNativeQuery(String sql) {
121            return this.em.createNativeQuery(sql, type).getResultList()
122        }
123 
124        /**
125         * Returns the number of total records
126         * @param namedQueryName
127         * @return int
128         */
129        public int countTotalRecord(String namedQueryName) {
130            Query query = em.createNamedQuery(namedQueryName);
131            Number result = (Number) query.getSingleResult();
132            return result.intValue();
133        }
134 
135        /**
136         * Returns the number of records that meet the criteria with pa
137         * result limit
138         * @param namedQueryName
139         * @param parameters
140         * @param resultLimit
141         * @return List
142         */
143        public List findWithNamedQuery(String namedQueryName, Map param
144            Set<Map.Entry<String, Object>> rawParameters = parameters.e
145            Query query = this.em.createNamedQuery(namedQueryName);
146            if (resultLimit > 0) {
147                query.setMaxResults(resultLimit);
148            }
149            for (Map.Entry<String, Object> entry : rawParameters) {
150                query.setParameter(entry.getKey(), entry.getValue());
151            }
152            return query.getResultList();
153        }
154 
155        /**
156         * Returns the number of records that will be used with lazy lo
157         * @param namedQueryName
158         * @param start
159         * @param end
160         * @return List
161         */
162        public List findWithNamedQuery(String namedQueryName, int start
163            Query query = this.em.createNamedQuery(namedQueryName);
164            query.setMaxResults(end - start);
165            query.setFirstResult(start);
166            return query.getResultList();
167        }
168    }
```

The JSF page that loads user list, create, delete, and update users:

```
1   <ui:composition template="/templates/layout.xhtml"
```

```
    xmlns="http://www.w3.org/1999/xhtml"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:p="http://primefaces.org/ui">
    <ui:define name="content">
        <h:form id="form" onLoad="reset()">
                <p:dataTable id="dataTable" var="user" value="#{use
                        paginator="true" rows="10" selection="#{use
                        paginatorTemplate="{CurrentPageReport}  {Fi
                        lazy="true" rowsPerPageTemplate="10,15,50">
                <f:facet name="header">
                    User List
                </f:facet>
                <p:column selectionMode="multiple" style="width:18p
                <p:column>
                    <f:facet name="header">
                        <h:outputText value="Username" />
                    </f:facet>
                    <p:commandLink value="#{user.username}" update=
                        <f:setPropertyActionListener value="#{user}
                    </p:commandLink>
                </p:column>

                <p:column>
                    <f:facet name="header">
                        <h:outputText value="Firstname" />
                    </f:facet>
                    <h:outputText value="#{user.firstname}" />
                </p:column>

                <p:column>
                    <f:facet name="header">
                        <h:outputText value="Lastname" />
                    </f:facet>
                    <h:outputText value="#{user.lastname}" />
                </p:column>

                <p:column>
                    <f:facet name="header">
                        <h:outputText value="Email" />
                    </f:facet>
                    <h:outputText value="#{user.email}" />
                </p:column>
               <!--  <p:column>
                    <f:facet name="header">
                        <h:outputText value="Role name" />
                    </f:facet>
                    <h:outputText value="#{user.role.rolename}" />
                 </p:column> -->
               <f:facet name="footer">
                    <p:commandButton value="New User" oncomplete="n
                    <p:commandButton value="Delete Users" actionLis
                 </f:facet>
            </p:dataTable>
        </h:form>

            <p:dialog header="User Detail" widgetVar="userDialog"
              <h:form id="userDetailForm">
                <p:panelGrid id="display" columns="2" cellpadding="
            <h:outputText value="Username :"></h:outputText>
                        <h:outputText value="#{userController.selec

            <h:outputText value="First name :"></h:outputText>
                        <h:inputText value="#{userController.select

            <h:outputText value="Last name :"></h:outputText>
                        <h:inputText value="#{userController.select

            <h:outputText value="Email :"></h:outputText>
            <h:inputText value="#{userController.selectedUser.email

                        <h:outputText value="Role :"></h:outputText
                        <p:selectManyMenu  id="newUserRole" require
                            <f:selectItems  value="#{userController
                            <f:converter converterId="com.nz.util.O
                        </p:selectManyMenu>

            <h:outputText value="Street Name :"></h:outputText>
            <h:inputText value="#{userController.selectedUser.addre

            <h:outputText value="Suburb :"></h:outputText>
            <h:inputText value="#{userController.selectedUser.addre

            <h:outputText value="City :"></h:outputText>
            <h:inputText value="#{userController.selectedUser.addre

            <h:outputText value="Country :"></h:outputText>
            <h:inputText value="#{userController.selectedUser.addre
```

```
                    <f:facet name="footer">
                            <p:commandButton value="Update" update=
                    </f:facet>
                </p:panelGrid>
            </h:form>
        </p:dialog>

        <p:dialog header="Create New User" widgetVar="newUserDial
            <h:form id="newUserForm">
              <p:panelGrid id="displayNewUser" columns="2" cellpa
          <h:outputText value="Username :"></h:outputText>
                    <p:inputText value="#{userController.newUse

          <h:outputText value="First name :"></h:outputText>
                    <p:inputText value="#{userController.newUse

          <h:outputText value="Last name :"></h:outputText>
                    <p:inputText value="#{userController.newUse

                    <h:outputText value="Password :"></h:output
                    <p:inputText value="#{userController.newUse
                        <f:converter converterId="com.nz.util.S
                    </p:inputText>

          <h:outputText value="Email :"></h:outputText>
          <p:inputText value="#{userController.newUser.email}"/>

                    <h:outputText value="Role :"></h:outputText
                     <p:selectManyMenu  id="newUserRole" requir
                         <f:selectItems  value="#{userControlle
                         <f:converter converterId="com.nz.util.
                     </p:selectManyMenu >
          <h:outputText value="Street Name :"></h:outputText>
          <p:inputText value="#{userController.newUser.address.st

          <h:outputText value="Suburb :"></h:outputText>
          <p:inputText value="#{userController.newUser.address.su

          <h:outputText value="City :"></h:outputText>
          <p:inputText value="#{userController.newUser.address.ci

          <h:outputText value="Country :"></h:outputText>
          <p:inputText value="#{userController.newUser.address.co
                  <f:facet name="footer">
                          <p:commandButton value="Submit" update=
                          <p:commandButton type="reset" value="Re
                  </f:facet>
                </p:panelGrid>
              </h:form>
          </p:dialog>

<p:growl id="growl" showDetail="true" life="5000" />

<script type="text/javascript">
    function handleSubmitRequest(xhr, status, args, dialogName, for
            dialog = jQuery('#'+dialogName);
        if(args.validationFailed) {
            dialog.effect("shake", { times:3 }, 100);
        } else {
            clearForm(formName);
            newUserDialog.hide();
            userDialog.hide();
        }
    }
    function clearForm(formName){
        jQuery('#'+formName).each(function(){
            this.reset();
    });
    }
</script>
     </ui:define>
</ui:composition>
```

The backing bean that is used in the web layer

```
@ManagedBean
@ViewScoped
public class UserController implements Serializable  {

        private @Inject DataAccessService das;
        private User[] selectedUsers;
        private LazyDataModel<User> lazyModel;
        private User newUser = new User();
        private User selectedUser = new User();
        private List<Role> roleList;

    public UserController() {

```

```
    }

        @PostConstruct
        public void init(){
            lazyModel = new LazyUserDataModel(das);
            roleList = das.findWithNamedQuery(Role.ALL);
        }

        public void doCreateUser() {
            das.create(newUser);
        }

        public void doUpdateUser(ActionEvent actionEvent){
            das.update(selectedUser);
        }

        public void doDeleteUsers(ActionEvent actionEvent){
            das.deleteItems(selectedUsers);
        }

        /*
         * Getters, Setters
         */

        public User getSelectedUser() {
            return selectedUser;
        }

        public void setSelectedUser(User selectedUser) {
            this.selectedUser = selectedUser;
        }

        public User[] getSelectedUsers() {
            return selectedUsers;
        }

        public void setSelectedUsers(User[] selectedUsers) {
            this.selectedUsers = selectedUsers;
        }

        public User getNewUser() {
            return newUser;
        }

        public void setNewUser(User newUser) {
            this.newUser = newUser;
        }

        public LazyDataModel<User> getLazyModel() {
            return lazyModel;
        }

        public List<Role> getRoleList() {
            return roleList;
        }

        public void setRoleList(List<Role> roleList) {
            this.roleList = roleList;
        }
}
```

I also have a custom converter that converts password to SHA-256 format

```
@FacesConverter("com.nz.util.SHAConverter")
public class SHAConverter implements Converter {

@Override
public Object getAsObject(FacesContext context, UIComponent componen

if(value.equalsIgnoreCase(""))
return "";
else if (value.length()&lt;4){
FacesContext.getCurrentInstance().addMessage("newPassword", new Face
return value;
}
try {
MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
byte[] hash = messageDigest.digest(value.getBytes("UTF-8"));
StringBuilder stringBuilder= new StringBuilder();
for (int i = 0; i &lt; hash.length; i++) {
stringBuilder.append(Integer.toString((hash[i] &amp; 0xff) + 0x100,
}
return stringBuilder.toString();
} catch (NoSuchAlgorithmException ex) {
Logger.getLogger(SHAConverter.class.getName()).log(Level.SEVERE, nul
return "";
} catch (UnsupportedEncodingException unsupportedEncodingException)
Logger.getLogger(SHAConverter.class.getName()).log(Level.SEVERE, nul
return "";
```

```
27  }
28  }
29
30  @Override
31  public String getAsString(FacesContext context, UIComponent componen
32  return "";
33  }
34  }
```

Download the source code here
Please try live demo (Username: Admin, Password:1234)

Have a good one!

---

28 thoughts on "Part 4 – Lazy Loading and Pagination with JSF 2.1 / Primefaces 3.4.2 DataTable and Form validation"

Reply

Jan 3, 2013 4:47 am

Pete

Hello – This is a great example. I like how PrimeFaces works here, but I'm confused on how the page flow is set up. Is it all just 1 JSF? Also, is there a way to log on to the demo as a manager and see a different view gor that security role? Thanks for the great tutorial!

Reply

Jan 3, 2013 8:27 am

simtay

Hey Pete,

Thanks for visiting my website. If you download the code and look at web.xml file you will see there are Security Constraints and a URL pattern specified for administrators. In our example, it is /admin/* which means that if you have the manager role you won't be able have access to any pages that are listed under admin folder/path. If you create a user that only has the manager or user role and try to log into the system with this username and password you provided (simtay.tk) you won't be able to see user list page. I simply used command button with action for the page flows ( ) hope that helps. If you have any more question please feel free to contact me or ask it here.
Thanks again...
Cheers Emre

Reply

Jan 13, 2013 7:35 am

Pete

Thanks Emre, that makes sense now. I have viewed around 10 similar tutorials on the web from various sources (including the Netbeans example pages) and this is by far the best tutorial because it uses the most current technologies like Maven rather than Ant, MySQL rather than just using the included Netbeans Java DB, it uses PrimeFaces rather than just stopping at standard JSF, and it includes security. Big step above all the other examples out there.

Pete

Reply

Jan 13, 2013 11:18 am

simtay

Hey Pete

You are welcome. Thanks for the compliment. If you have any suggestions for future topics, let me know.

Cheers

Reply

Jan 22, 2013 10:21 pm

Hjelperne

Great example.
Would it also be possible to post your namedQueries?

Reply

Jan 23, 2013 8:01 am

simtay

Thank you Hjelperne, please see http://www.simtay.com/part-3-basic-implementation-of-crud/ there is a section about Named Queries. Cheers Emre

Reply

Mar 15, 2013 6:28 am

Helio Frota

Good example of datatable and dialogs ! I will check thanks !

Reply

Mar 15, 2013 7:43 pm

simtay

Thanks Helio

Reply

May 24, 2013 5:41 am

Vusal

Emre, selam. Ben Azerbaicandan Vusal. Emre, ben bir wey ariyorum belki sen bana yardimci olursun. Ben bu sahede daha yeniyim.
Emre, benim veritabandan chekmek gerek bilgilerim var, amma bu bilgiler resultlist olaraq 1.000.000 entity geri donuyor ve bu da dogru duzgun chaliwmiyor. Bu kadar veriyi ben jsf sayfasinda datatable oluwduraraq ichine doldura bilmiyorum. Bana yarimdimci olursan cok sevinirim.

Reply

May 24, 2013 8:03 am

simtay

Merhaba Vusal!

Veri tabanindan bilgileri “Lazy Loading” teknigi kullanarak cekmen gerekli. Bu dorduncu kisimda nasil yapildigini cok detayli olmasada anlatmaya calistim. Hangi asamadasin? Nerde sorun yasiyorsun? Biraz daha detayli anlatirsan sorununu yardimci olurum. Bana email atabilirsin. Saygilar, Emre…

Reply

Jun 11, 2013 3:08 pm

kishore

Hi Simtay

This example is quite good. Is there any one with EJB3 integrated with JSF2 with hibernate?
thx

Reply

Jul 5, 2013 2:19 pm

simtay

Sorry not using EJB 3 only CDI, cheers

Reply

Jun 11, 2013 4:25 pm

Street Hawk

Simtay

Is this datamodel allow sorting for whole results or is it by page only?

Reply

Jul 5, 2013 2:18 pm

simtay

it's by page only, you can list it sorted using CriteriaQuery

public List findWithCriteriaQuery(String sortField, SortOrder sortOrder, int start, int end) {

CriteriaBuilder cb = em.getCriteriaBuilder();
CriteriaQuery cq = cb.createQuery();
Root e = cq.from(User.class);
if (sortField != null){
if (sortOrder == SortOrder.ASCENDING) {
cq.orderBy(cb.asc(cb.lower(e.get(sortField))));
} else {
cq.orderBy(cb.desc(cb.lower(e.get(sortField))));
}
}
Query query = em.createQuery(cq);
query.setMaxResults(end – start);
query.setFirstResult(start);
return query.getResultList();
}

Reply

Nov 21, 2013 11:23 pm

Farid

bu for me , i am using Hibernate (DAO).
How i can procedes,
Thank you

Reply

Nov 22, 2013 7:20 pm

simtay

I guess it should be easy, you should be able to change it to hibernate on persistence.xml file and you can use Hibernate JPA api. See what happens. I will also try and let you know. Cheers Emre

Reply

Jul 31, 2013 11:33 pm

Carl

Thanks for the article.

Just to make sure I've understood your response to the previous comment… There is no easy way for me to view the users with last names starting with 'z' (for example) because I can only sort one page at a time. Is this correct? This seems to be an issue with Primefaces rather than your code, but it's a pretty serious limitation. I've written my own load() method but it doesn't receive any sorting parameters from Primefaces after I've changed page. In other words, I sort the first page, then I move to page 2 and it is no longer sorted.

Reply

Aug 2, 2013 12:05 pm

simtay

Hi Carl, it is not an issue with PrimeFaces. I have given example how to sort with lazy loading implementation. You need to modify your code. Please look at my example see if you improve it. Cheers

public List findWithCriteriaQuery(String sortField, SortOrder sortOrder, int start, int end) {

CriteriaBuilder cb = em.getCriteriaBuilder();
CriteriaQuery cq = cb.createQuery();
Root e = cq.from(User.class);
if (sortField != null){
if (sortOrder == SortOrder.ASCENDING) {
cq.orderBy(cb.asc(cb.lower(e.get(sortField))));

```
} else {
cq.orderBy(cb.desc(cb.lower(e.get(sortField))));
}
}
Query query = em.createQuery(cq);
query.setMaxResults(end – start);
query.setFirstResult(start);
return query.getResultList();
}
```

Reply

Sep 16, 2013 11:21 pm

Halil Karakose

Hi Emre
I checked PrimeFaces repo but 3.5-SNAPSHOT does not exist. Instead there is a 3.5 version dependency

Reply

Sep 17, 2013 12:21 pm

simtay

Alright, I will check it later.

Reply

Nov 8, 2013 11:03 am

maihacke

How do you deal with the problem that LazyDataModel has to be serializable, but contains a reference to DataAccessService which is not?

Reply

Nov 12, 2013 9:15 pm

simtay

It doesn't have to be. Cheers

Reply

Nov 12, 2013 9:09 pm

Prax

Hi Simtay,

Very very good tutorial provided. It really helps.

I have question regarding **LazyDataModel** and **Live Scrolling**.

I wanted to know how does both works. Does Live Scroll load all data together in cache? If it loads all data in cache then It doesn't seems to be good idea. Boss says please implement live scroll as user wants replacing LazyDataModel. I have implemented mock pages with 50K records of Live Scrolling and seems faster too as we scroll down rather visiting each pages.

Please suggest.

Prax.

Reply

Nov 12, 2013 9:14 pm

simtay

Yes, with lazyDataModel you can do pagination, only get certain numbers of records, not all of them. Please go have a look at the example on Primefaces webpage.

Reply

Nov 12, 2013 10:23 pm

Prax

For lazy loading lets say there are 200 pages and I visit one by one 100 pages and then I go on previous pages, does this reload the data each time or 100 pages records it keep

in cache?

Reply

Nov 13, 2013 9:03 pm

simtay

I will reload 100 pages again every time you change the page, but it will use the caching mechanism if nothing has been updated.

Reply

Dec 6, 2013 6:43 am

Mohamed

Hi smitay nice example ,
just one problem : if i selected items from the first page and then navigating to the second page to select others, i guess everything gets loaded again and the selected items on the first page are not selected anymore + the ones selected on the second page two

is there a better solution than going with rowSelect or rowSelectCheckbox ajax events to remember all the selected items

Reply

Dec 6, 2013 9:21 pm

simtay

Hey thanks mate, I guess you can ask it on primefaces forum page. I remember I did look for a solution but I couldn't find one. There could be a fix for that problem now but I cannot guarantee it you might need to do a hack!

Leave a Reply

Author (required)

Email (will not be published)(required)

Website

Just to prove you are a human, please answer the following math challenge either out loud or typing into the text box
seven × = sixty three

b i link b-quote code close tags

Post Comment

« Part 3 – Basic Implementation of CRUD
Part 5 – Downloading PrimeFaces themes / creating and installing custom themes and overriding PrimeFaces style sheets. »