Backing Bean

Fetch Unique NSF Attachments

Xpages

System Requirements:
Download Domino Designer 8.5.2 Environment (DDE)
http://www.ibm.com/developerworks/downloads/ls/dominodesigner/

Introduction:
Build a connection to a Lotus Notes Domino Database located on C Drive using a Backing Bean, supplies Xpages form list of attachments/images to choose from, click unique image to query NSF back-end.

Disclaimer:
Information contained in the following is presented as is. This tutorial assumes you have basic programming knowledge. All tutorials are based on an Eclipse/Eclipse-based software. Should you need to familiarize yourself with a certain Eclipse environment, prior to continuing this tutorial, please stop now and see our Tutorials page...

Build Unique Attachment Results into Xpages
At this point we assume Domino Designer 8.5.2 is downloaded/installed and the Backing Bean and Xpage are already created. Copy and paste below code samples to your environment.

RetrieveNewNoticeBackingBean.java

```
/**
 * Copyright 2012 Dököll Solutions, Inc.
   Licensed under the Apache License, Version 2.0 (the "License");
   you may not use this file except in compliance with the License.
   You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0

   Unless required by applicable law or agreed to in writing, software
   distributed under the License is distributed on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
   See the License for the specific language governing permissions and
   limitations under the License.

 * Program: RetrieveNewNoticeBackingBean.java
 * Created from Copy: 2012.06.11.11.05.PM
 * New Retrieval Bean for Toy Submissions via Xpages
 *
 */
package com.osc.its.bulletin.board.JavaBeans;
```

```
//...
//java imports
import java.util.Vector;
//faces imports
import javax.faces.context.FacesContext;
//servlet imports
import javax.servlet.http.HttpServletRequest;
//notes imports...
import lotus.domino.Item;
import lotus.domino.View;
import lotus.domino.local.Database;
import lotus.domino.local.Document;


/**
 * @author Dököll Solutions, Inc.
 * @version 2012.06.11.11.05.PM
 *
 */


public class RetrieveNewNoticeBackingBean {

      // Declare variable, load view name
      final String ViewName = "By Category";

      // Get Keyword collection
      @SuppressWarnings( { "null", "unchecked" })
      public Keyword[] getKeywords() {

            // Declare Keyword Array
            Keyword[] keywords = null;

            try {

                  // get the current database being used
                  Database database = (Database) FacesContext.getCurrentInstance()
                              .getApplication().getVariableResolver().resolveVariab
le(
                  FacesContext.getCurrentInstance(), "database");


                  // Get the full URL and join it with the current document
                  // from Notes back-end
                  // 2012.03.02.8.47.PM
                  HttpServletRequest req = (HttpServletRequest) FacesContext
                              .getCurrentInstance().getExternalContext().getRequest
();

                  //...
                  //Declare and variable to grab full URL
                  String baseURL =
req.getRequestURL().toString().replace(
                  req.getRequestURI(), req.getContextPath());

                  // Find the view in question
```

```
                    View view = database.getView(ViewName);
                    //load to console for debugging purposes
                    System.out.println("View Obtained..." + view);

                    //...
                    Document sDoc;
                    //...
                    Document ndoc;

                    //grab our first doc
                    sDoc = (Document) view.getFirstDocument();
                    //load to console for debugging purposes
                    System.out.println("Document Obtained..." + sDoc);


                    // load documents count
                    keywords = new Keyword[view.getEntryCount()];


                    // Run through Keyword document collection
                    int docount = 0;
                    while (sDoc != null) {

                          //initialize vector, load items
                          Vector<Item> itemVector = sDoc.getItems();

                          //run though and pull off items (attachments) one at a time
                          for (Item inboundItem : itemVector) {
                                //ensure item is of type attachment
                                if (inboundItem.getType() == Item.ATTACHMENT) {
                                      //declare and initialize attachment placeholder
          String AttMents = inboundItem.getValueString();

                                      //call Keyword and fill Category, Topic,
BrowserURL, FileName variables...
                                      Keyword keyword = new Keyword(null, null, null,
null);

                                      // render document(s) to variables of Keyword
object
                                      keyword.setCategory(sDoc
                                                  .getItemValueString("Categories"));

      keyword.setTopic(sDoc.getItemValueString("Subject"));
                                      keyword.setBrowserURL(baseURL + "/" + view +
"/"
                                                  + sDoc.getUniversalID() + "?
OpenDocument");
          keyword.setFileName(baseURL + "/" + view + "/"
          + sDoc.getUniversalID() + "/" + "$FILE/"
          + sDoc.getAttachment(AttMents));//

                                      // Send Keyword object into Keywords Array
                                      keywords[docount] = keyword;
                                      // increment counts
                                      docount += 1;
```

```
                                    // load next doc
                                    ndoc = (Document) view.getNextDocument(sDoc);
                                    sDoc = ndoc;
                              }
                        }

                  }
                  // return the Array
                  return keywords;


                  //catch accordingly
            } catch (Exception e) {
                  e.printStackTrace();
            }
            // return nothing
            return null;

      }


}
```

Build Xpage file

Full code added below, jump ahead if necessary, but do read the code, we will be opening attachments one by one and allow unique documents to be viewed. Areas of interest have been highlighted for your convenience.

xpbycategory.xsp

```
<?xml version="1.0" encoding="UTF-8"?>
<xp:view xmlns:xp="http://www.ibm.com/xsp/core">
      &#160;

      <xp:dataTable id="dataTable1" var="categoryTable"
            value="#{javascript:RetrieveNewNoticeBackingBean.getKeywords()}"
            style="width:80.0%;background-color:rgb(220,237,237)"
rows="7"><xp:this.facets>
                  <xp:pager partialRefresh="true" layout="FirstImage PreviousImage
SeparatorPage Group NextImage LastImage" xp:key="header" id="pager1"
for="dataTable1" style="color:rgb(0,64,64);font-weight:bold;background-
position:top center" />

                  <xp:pager partialRefresh="true" layout="FirstImage PreviousImage
SeparatorPage Group NextImage LastImage" xp:key="footer" id="pager2"
for="dataTable1" title="By Category" style="color:rgb(0,64,64);font-
weight:bold;background-position:top center" />

            </xp:this.facets><xp:column id="column3">
                  <xp:link escape="true" id="link1"
                        style="text-align:center;background-position:center
center;width:88.0px">
                        <xp:this.value>
                              <![CDATA[#{javascript:categoryTable.browserURL}]]>
                        </xp:this.value>


                        <xp:image id="image1" url="/search.gif" alt="View File"
                              style="align:center" />

                  </xp:link><xp:this.facets>
                        <xp:label id="label1" xp:key="header" style="font-
weight:bold;font-family:Arial Black;font-size:14pt;background-
color:rgb(220,237,237);color:rgb(0,64,0)" for="image1" value="File" />

                  </xp:this.facets>

            </xp:column>

<xp:column id="column4"><xp:this.facets><xp:label value="Type" id="label3"
xp:key="header" style="font-weight:bold;font-family:Arial Black;font-
size:14pt;background-color:rgb(220,237,237);color:rgb(0,64,0)" /></xp:this.facets>
<xp:link escape="true" text="#{categoryTable.topic}" id="link2"
                        value="#{javascript:categoryTable.browserURL}"
                        title="Type" style="color:rgb(0,0,64);font-weight:bold"/>
</xp:column>

            <xp:column id="column2">
                  <xp:this.facets>
                        <xp:label value="Category" id="category1"
                              xp:key="header"
                              style="font-weight:bold;font-family:Arial Black;font-
size:14pt;background-color:rgb(220,237,237);color:rgb(0,64,0)" />

                  </xp:this.facets>
                  <xp:text escape="true" id="computedField2"
                  value="#{categoryTable.category}" style="color:rgb(64,0,0)" />

            </xp:column>

            <xp:column id="column1"><xp:this.facets><xp:label value="Image"
id="label2" xp:key="header" style="font-weight:bold;font-family:Arial Black;font-
size:14pt;background-color:rgb(220,237,237);color:rgb(0,64,0)" /></xp:this.facets>
                  <xp:link escape="true" id="link4"
                        style="text-align:center;background-position:center
center;width:88.0px">
                        <xp:this.value>
                              <!
[CDATA[#{javascript:categoryTable.fileName}]]>
                        </xp:this.value>


                        <xp:image id="image3"

      url="#{javascript:categoryTable.fileName}" alt="View
File"
                              style="align:center;height:38.0px;width:38.0px" />

                  </xp:link>

            </xp:column>
      </xp:dataTable>

</xp:view>
```

Conclusion:
You can now retrieve a list of images from NSF back-end and further search the back-end using unique image results.

Added info: You will need to reference `RetrieveNewNoticeBackingBean.java` in your faces-config.xml file.

Questions, comments, please post a brief message on our Contact form on the main site.

Thank you for coming...