Döcu Content
Phantom Pages and Links
JavaBean
Xpages Controls Per User Access

Introduction:
Xpages forms and links rendering have never been anymore dynamic. Load Documents, pages and so on via links on Xpages form, deposited to form on the fly, based on user permissions. Page names are submitted to back-end by each user to be called by Döcu Content App and allow user to click on own/specific link to get to page of choice.

Disclaimer:
Information contained in the following is presented as is. This tutorial assumes you have basic Lotus Notes Configuration and Programming knowledge.

Döcu Content JavaBean
Create JavaBean to read NSF back-end and grab page names added through Admin pages with the intent of using these page names through link controls to be viewed. These pages would have been added by the curent General user logged on or an Admin user inserting on other user's behalf. Each user has own page, thus link controls will list the specific page name to click on to get to that page. We recommend watching the following Youtube video for additional information:
http://www.youtube.com/watch?v=OOMpimt8X30&feature=c4-overview&list=UUSImDTpK0oe7QrPsYOE4nww

JavaBean Create Instructions, see screenshots for areas of interest

1. Create new JavaBean to read NSF back-end view documents
2. Load Default Error page, where user does not have a page a retrieve

```
120    public String getAddURL() {
121        //throw user's default page link to form's Navigation
122        //provided user has the page, previous submitted to back-end
123        //as preferences
124        if (UrlID.equalsIgnoreCase(RoleNameValue)) {
125            System.out.println("User Access Okay() for..." + AddURL);
126            return AddURL;
127            //give user a default error-friendly page
128            //TODO: Put some stuff on there, give user options to try
129            //      Put some items user can do/see, as default information
130        } else {
131            System.out.println("User Access for User..."
132                    + "No Page URL Found");
133            return "/xpnopageaccess.xsp";
134        }
135
136    }
```

PhantomPageBean.java;

```
/**
 * Copyright 2014 Dökoll 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: PhantomPageBean.java
 * @AppName: DöcuContent.nsf
 * Created: 2014.03.01.6.12.PM
 * JavaBean Class to load User-Specific Pages based on permissions, and URL
Security Page submissions
 * these pages would have been submitted to NSF back-end either by an Admin or
the User
 */
package com.dokoll.solutions.inc.developement.Utils;

//faces imports
import javax.faces.context.FacesContext;
//servlet imports
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServletRequest;
//domino imports
import lotus.domino.Database;
import lotus.domino.NotesException;
import lotus.domino.Session;
import lotus.domino.View;
import lotus.domino.ViewEntry;
import lotus.domino.ViewEntryCollection;
import com.ibm.xsp.model.domino.DominoUtils;

/**
 * @author Dökoll Solutions, Inc.
 * @version: 2014.03.01.6.12.PM
 */

public class PhantomPageBean {

	// ...
	// declare static variable for comparison block
	static String URL_SEC_ROLE_VIEW = "SecurityAccessView";
	static String URL_SEC_ACCESS_VIEW = "MyURLSecurityAccessView";
	static String URL_SEC_PAGE_NODEFAULT = "No Default Page";

	// ...
	// 2014.03.01.6.19.PM
	public String UserName;
	// ...
	// DöcuContent data
	public String UrlID=null;
	public String ViewURL=null;
	public String AddURL=null;
```

```
public String UpdateURL=null;
public String DeleteURL=null;
// ...
// Cookie data
public String UserNameValue;
public String RoleNameValue;

// getter/setter methods

/**
 * @return the userName
 */
public String getUserName() {
      return UserName;
}

/**
 * @param userName
 *            the userName to set
 */
public void setUserName(String userName) {
      UserName = userName;
}

/**
 * @return the urlID
 */
public String getUrlID() {

      return UrlID;
}

/**
 * @param urlID
 *            the urlID to set
 */
public void setUrlID(String urlID) {
      UrlID = urlID;
}

/**
 * @return the viewURL
 */
public String getViewURL() {
      return ViewURL;
}

/**
 * @param viewURL
 *            the viewURL to set
 */
public void setViewURL(String viewURL) {
      ViewURL = viewURL;
}

/**
 * @return the addURL
 */
public String getAddURL() {
      //throw user's default page link to form's Navigation
      //provided user has the page, previous submitted to back-end
      //as preferences
      if (UrlID.equalsIgnoreCase(RoleNameValue)) {
```

```
                System.out.println("User Access Okay()
for..." + AddURL);
                return AddURL;
                //give user a default error-friendly page
                //TODO: Put some stuff on there, give user
options to try
                //      Put some items user can do/see, as
default information
           } else {
                System.out.println("User Access for User..."
                        + "No Page URL Found");
                return "/xpnopageaccess.xsp";
           }

      }

      /**
       * @param addURL
       *            the addURL to set
       */
      public void setAddURL(String addURL) {
            AddURL = addURL;
      }

      /**
       * @return the updateURL
       */
      public String getUpdateURL() {
            return UpdateURL;
      }

      /**
       * @param updateURL
       *            the updateURL to set
       */
      public void setUpdateURL(String updateURL) {
            UpdateURL = updateURL;
      }

      /**
       * @return the deleteURL
       */
      public String getDeleteURL() {
            return DeleteURL;
      }

      /**
       * @param deleteURL
       *            the deleteURL to set
       */
      public void setDeleteURL(String deleteURL) {
            DeleteURL = deleteURL;
      }

      /**
       * @return the userNameValue
       */
      public String getUserNameValue() {
            return UserNameValue;
```

```
	}

	/**
	 * @param userNameValue
	 *            the userNameValue to set
	 */
	public void setUserNameValue(String userNameValue) {
		UserNameValue = userNameValue;
	}

	/**
	 * @return the roleNameValue
	 */
	public String getRoleNameValue() {
		return RoleNameValue;
	}

	/**
	 * @param roleNameValue
	 *            the roleNameValue to set
	 */
	public void setRoleNameValue(String roleNameValue) {
		RoleNameValue = roleNameValue;
	}

	public PhantomPageBean() {

		// get userCookies
		FacesContext facesContext = FacesContext.getCurrentInstance();
		String cookieName = null;
		Cookie cookie[] = ((HttpServletRequest) facesContext
				.getExternalContext().getRequest()).getCookies();
	if (cookie != null && cookie.length > 0) {
		for (int i = 0; i < cookie.length; i++) {
			cookieName = cookie[i].getName();
			if (cookieName.equals("cookieKeyUser")) {
			UserNameValue = cookie[i].getValue();
	System.out.println("WOAAAHHH! Found this
UserNameValue Cookie..."+ UserNameValue);
				}
		if (cookieName.equals("cookieKeyRole")) {
			RoleNameValue = cookie[i].getValue();
			System.out.println("WOAAAHHH! Found this
RoleNameValue Cookie..."+ RoleNameValue);
				} else
	System.out.println("Cookies not found...");

				// TODO: Add this method to JSFUtil class,
				// also delete the cookies when logging out
			}

		}
	}

	/**
	 *
	 * Method: doLoadPanthomPages() Created from Copy: 2014.03.01.6.30.PM URL
	 * Security Access Pages, some may be Hidden/Showing
	 *
	 */
```

```
	// links control, items on form load based on user url security pages
	// previously submitted
	// for viewurl, updateurl, addurl, and deleteurl in NSF back-end
	public void doLoadPanthomPages() {

		// let's add a try catch here, to grab errors near the end
		try {

			// BEGIN DEBUG
			System.out.println("BEGIN DEBUG | PhantomPageBean.java -
doLoadPanthomPages started... ");
			Session session = DominoUtils.getCurrentSession(FacesContext
					.getCurrentInstance());
			System.out.println("docucontent.nsf | doLoadPanthomPages() in
PhantomPageBean.java Session Obtained..."
					+ session);
			// find database in question from current session
			System.out.println("docucontent.nsf | doLoadPanthomPages() in
PhantomPageBean.java finding docucontent.nsf based on Global User Session...");
			Database database =
session.getDatabase(session.getServerName(),
					"docucontent.nsf");
			System.out.println("doLoadPanthomPages() in
PhantomPageBean.java Fetched Database..."
					+ database);
			System.out.println("Getting a database connection... ");

			System.out.println("PhantomPageBean.java | doLoadPanthomPages
Connected to "
					+ database);
			// Find the view in question
			View view = database.getView(URL_SEC_ACCESS_VIEW);
			System.out.println("PhantomPageBean.java | doLoadPanthomPages
Connected to "
					+ view);
			// determine database state for searching
			if (database.isFTIndexed())
				database.updateFTIndex(false);
			else
				database.updateFTIndex(true);

			// Collect entries based on search criteria in UserNameValue =
?
			// (Cookie/Session based values)
			ViewEntryCollection vec =
view.getAllEntriesByKey(UserNameValue);

			System.out.println("PhantomPageBean.java | ViewEntryCollection
Counts Obtained...");
			System.out.println("PhantomPageBean.java | ViewEntries "
					+ vec.getCount() + " Account(s)");
			System.out.println("PhantomPageBean.java | ViewEntryCollection
completed successfully...");

			ViewEntry tmpentry;
			ViewEntry entry = vec.getFirstEntry();

			// if so, load text/err Message(s) to user
			if (entry == null) {
```

```
                        // add Message value...
                        // TODO: Do something else with the current component
                        // example, combine with built-in items
                        // ...
                        UserName = "Anonymous";
                        UrlID = "General";
                        ViewURL = URL_SEC_PAGE_NODEFAULT;
                        AddURL = "xpnopageaccess.xsp";
                        System.out.println("No Access to Page");

      // otherwise load the value needed to let user continue
                  } else {

      // add value needed for pairing with cookies, and so on...
      String strUserName = entry.getDocument().getItemValueString("userID");
      String strUrlID = entry.getDocument().getItemValueString("URLID");

      //if user returned info is solid, be nice and let user see his or her
stuff
      //in which case, we're speaking of default pages previously submitted
      //These would have been submitted by either the user upon registration via
xpmyusersecuredefinedurl.xsp
      //or by an Admin, entering on behalf of users
      //TODO: Limit the powers of LoggedIn User who enters info, based on
permissions...
      // Only Admins should be able to ad stuff for others
      if
(strUserName.equalsIgnoreCase(UserNameValue)
&& strUrlID.equalsIgnoreCase(RoleNameValue))
{

            UserName = entry.getDocument().getItemValueString("userID");
                        //DEBUG: Remove System.outs here, meant for
testing returned values from NSF back-end
            System.out.println("ENDING DEBUG | PhantomPageBean.java -
doLoadPanthomPages accessed: "+ UserName + " for UserName Field(s) / Link
controls... ");
            UrlID = entry.getDocument().getItemValueString("URLID");
                        //DEBUG: Remove System.outs here, meant for
testing returned values from NSF back-end
            System.out.println("ENDING DEBUG | PhantomPageBean.java -
doLoadPanthomPages accessed: "+ UrlID + " for User URL Field(s) / Link
controls... ");
            ViewURL = entry.getDocument().getItemValueString("ViewURL");
                        //DEBUG: Remove System.outs here, meant for
testing returned values from NSF back-end
            System.out.println("ENDING DEBUG | PhantomPageBean.java -
doLoadPanthomPages accessed: "+ ViewURL + " for User ViewURL Name Field(s) /
Link controls... ");
                  AddURL = entry.getDocument().getItemValueString("AddURL");
                        //DEBUG: Remove System.outs here, meant for
testing returned values from NSF back-end
            System.out.println("ENDING DEBUG | PhantomPageBean.java -
doLoadPanthomPages accessed: "+ AddURL + " for User PageName Field(s) / Link
controls... ");

            // doLoadPanthomAccess();
            tmpentry = vec.getNextEntry();
            entry.recycle();
```

```
                entry = tmpentry;

                System.out.println("DEBUG ENDED | PhantomPageBean.java -
doLoadPanthomPages released access to controls... ");
                return;
                // END DEBUG
                            }
                      }

                } catch (NotesException e) {
                      System.out.println(e.id + " " + e.text);
                      e.printStackTrace();

                } catch (Exception e) {

                      e.printStackTrace();
                }
                return;
          }

}
```

xpnavlinks.xsp Link Control Excerpt:

```
<xp:link escape="true" text="#{PhantomPageBean.viewURL}"
    id="addURL1" value="#{PhantomPageBean.addURL}" target="colours">
</xp:link>
```

View Entries for Each User:

1. **Valid users are AlienHouse and username101, should see own page**
2. **username102 being used for this exercise should see default page**

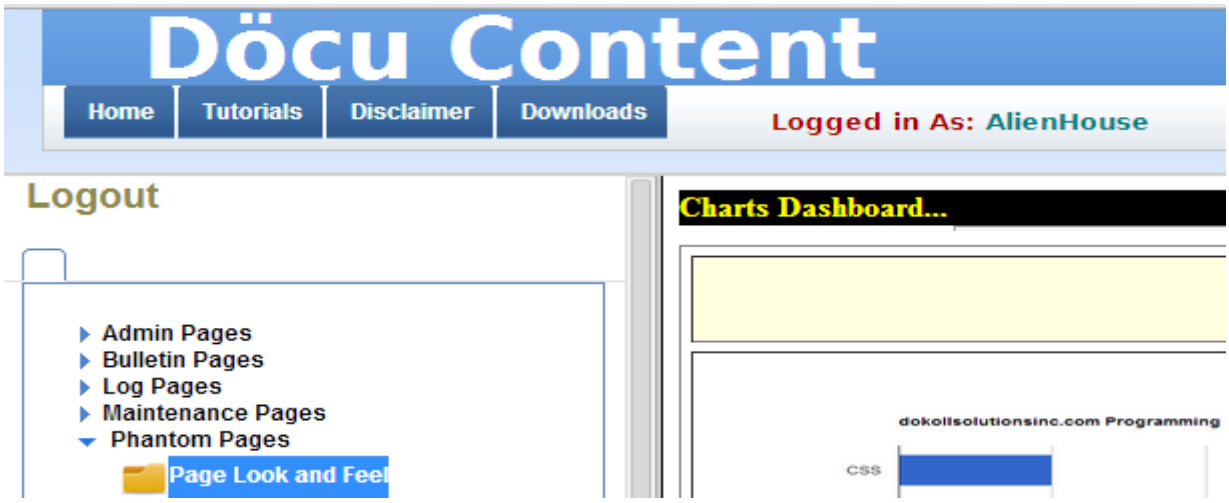
Home | docucontent.nsf - Views | MyURLSecurityAccessView - View

UserID	URLID	ViewURL	AddURL
AlienHouse	Admin	Page Look and Feel	xplookandfeel.xsp
username101	Tester	User-Defined Pages	xpuserdefinedpages.xsp
username107	Dev	User-Defined URLs	xpuserdefinedurl.xsp

AlienHouse User Screenshots:

User is Logged In

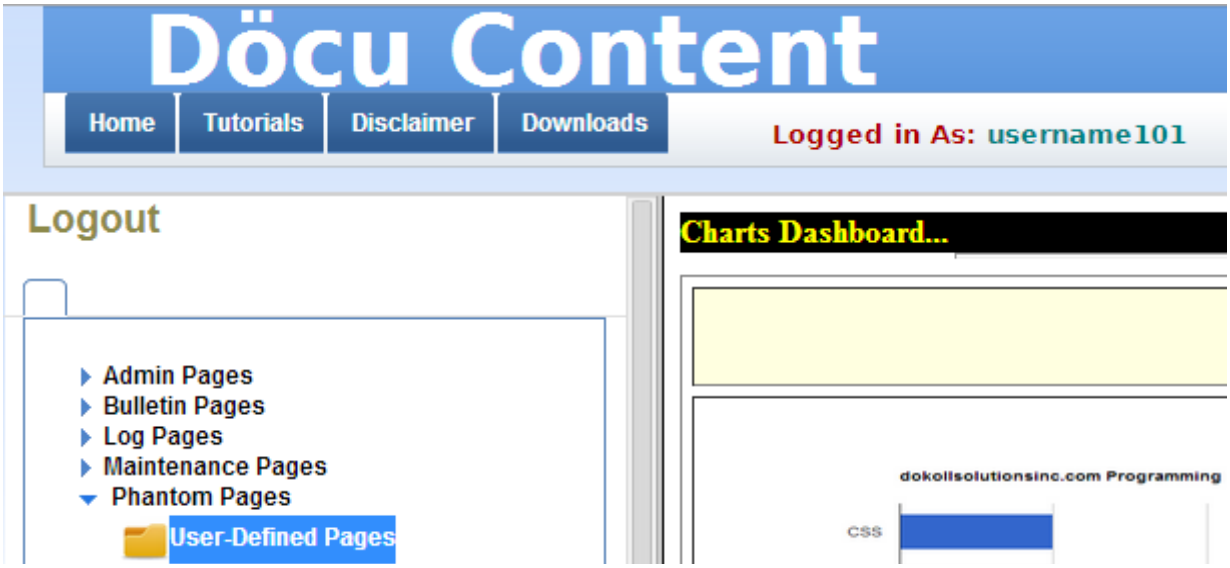
Here is the User Page, based on view item(s)



username101 User Screenshots:

User is Logged In

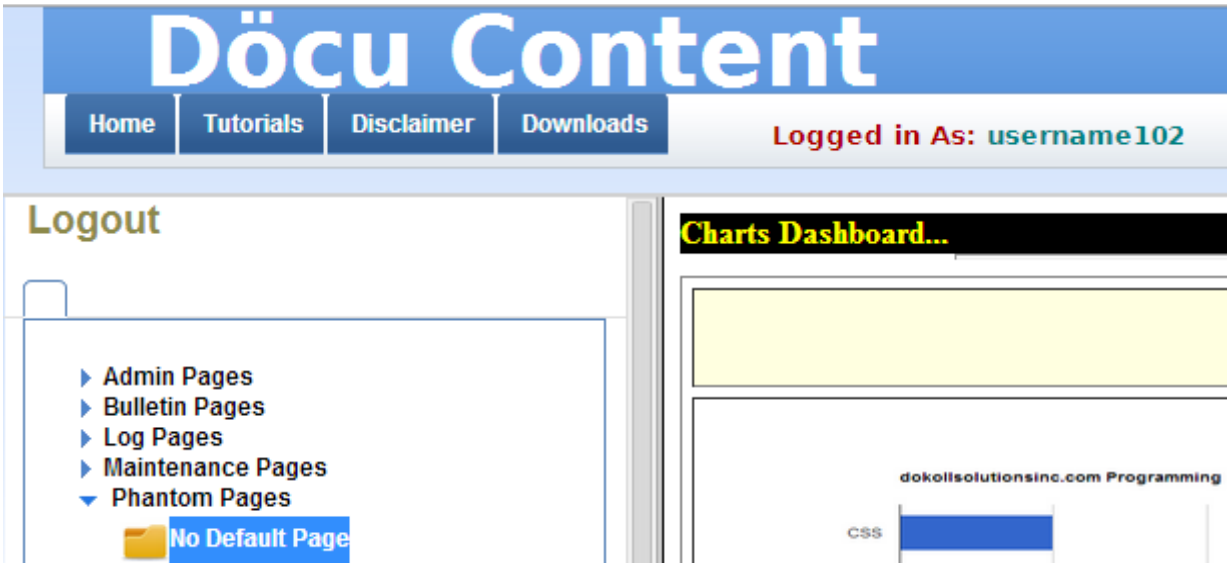
Here is the User Page, based on view item(s)



username102 User Screenshots:

User is Logged In

Here is the User Page, based on view item(s)



Conclusion:

You can now load specific page name and allow user to interact with preferences, based on user permissions. This should facilitate user-setting pages that only certain users need to see.

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

Thank you for coming...

Version:2014.03.02.11.00.PM