Dököll Solutions, Inc.

Application Development

Project Name:	Web News | Döcu Content	**Start Date:**	2016.08.22.10.21.PM
Purpose:	Instructions/Training	**End Date:**	2016.08.22.10.21.PM
Software:	IBM DB2, Notes Domino, JDeveloper	**Environment:**	Windows 7, 8, 10
Employee Name:	Dököll Solutions	**Employee ID:**	Dököll Solutions
Task(s):	JDeveloper JSF, Notes Domino Xpages	**Document:**	Journal Entries

JDeveloper JSF, Notes Domino Xpages

Foreword: Steps included below to create Apps via Oracle JDeveloper with IBM DB2 database, to interact with IBM Lotus Notes Domino on NSF back-end are current, however we suggest visiting Oracle and IBM websites for additional information.

System Requirements:

- Windows 7, 8, 10 Operating System Compatible
- Download IBM DB2 Express-C 9.x, 10.x, 11.x (not the Trial Version)
- http://www.ibm.com/developerworks/downloads/im/db2express/
- Download IBM Data Studio 4.x
- http://www.ibm.com/developerworks/downloads/im/data/ Download IBM Lotus Notes
- Domino Designer 8.5.3, 9.x
- https://www.ibm.com/developerworks/downloads/ls/dominodesigner/ Download Oracle
- JDeveloper 12.x
- http://www.oracle.com/technetwork/developer-tools/jdev/downloads/index.html

Disclaimer:
Information contained in this documentation is presented as is, we assume you are familiar with IBM DB2, Lotus Notes Domino Designer and Oracle JDeveloper. Rest assured, if you need additional support, you can always contact us or visit our website: www.dokollsolutionsinc.com for Free, ready to use, Step by Step PDF or YouTube Video Tutorials; if you would rather use your favourite search engine for help, we advise you to do so...

Introduction:
Create two sample Apps via Oracle JDeveloper and IBM Lotus Notes Domino Designer, run them side by side to communicate via XML file available in a URL. You will be using code below to help

you along (JavaBean, JavaAgent, Backing Bean, ADF/JSF and Xpages), areas of interest will be highlighted for your convenience.

Step 1

At this point, you should have gone through System requirements above- We suspect you already downloaded software and configured your environments; we are going to skip formal steps. We urge you to consult our documentation for support. Let's begin by setting you up for *success* with the samples in this tutorial, don't want to leave you guessing...

Hint # 1: If you downloaded IBM DB2 Express-C for this tutorial, you should be aware version 9.x that comes with Control Center is deprecated, you will need to download Data Studio- Perhaps you already have that version (9.x), rest assured you can still use it, powerful database application.

Hint # 2: If you downloaded IBM Data Studio 4.x or and earlier version separately for this tutorial, what a treat! However, you should know configuration can be a little getting used to. Please visit our website, Google, or use preferred search engine for additional support.

Hint # 3: No known issues reported with IBM Notes Domino Designer... if by any chance your Java code is faulty and your App is stuck, kill this bad boy via Task Manager, or reboot your system. You may also need to Clean your project in the IDE.

Hint # 4: No real issues reported with Oracle JDeveloper, except that it is helpful to run WebLogic Server prior to launching your Apps. If errors occur in your JSF pages, try to Rebuild the App in question, then you should be good to go, normal stuff-

Alright, enough Spoon-feeding on this front...

What to Expect:
The current consists of one JSF and one Backing Bean, with an XML file in the middle. Since you've done this before, there's a likelihood that you launched WebLogic Server and DB2 Database is running...

Additional Spoon-feeding *not required* here, this is a small App... However, you should still read through the information provided, then copy and paste the following to your environment

Here is the JSF Code: Once you have copied both the JSF and Backing Bean code and they are installed in your environment, you should first make sure WebLogic has finished starting up, then run the JSF page in Browser and get to work

webnewsdbfeeds.jsf

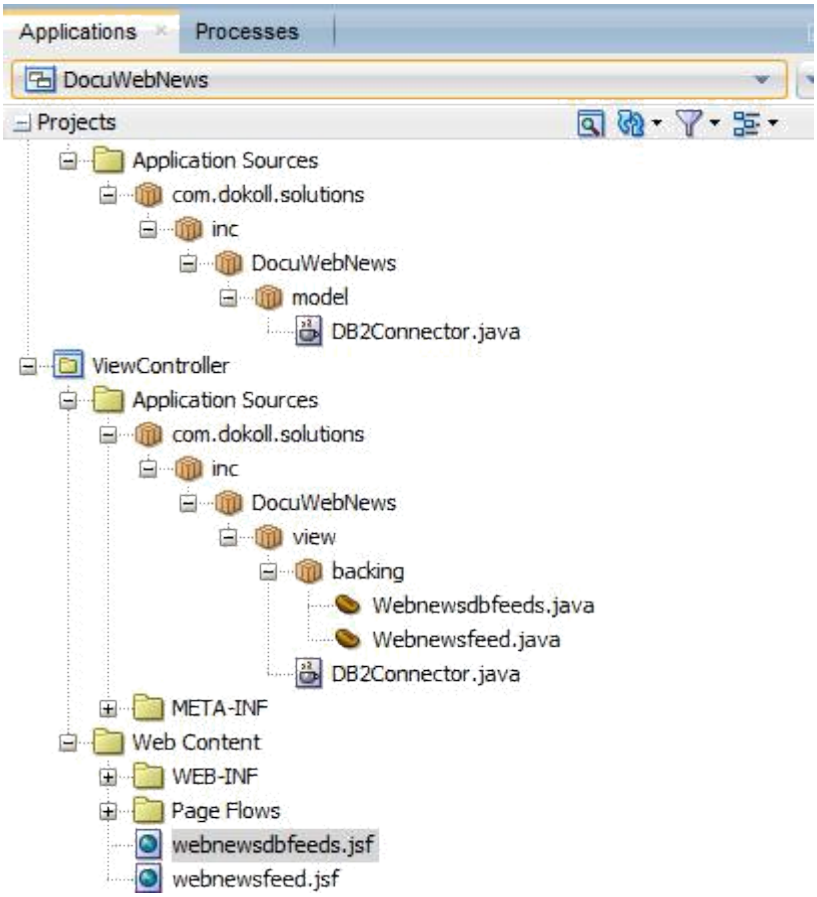
```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE html>
<f:view xmlns:f="http://java.sun.com/jsf/core"
xmlns:af="http://xmlns.oracle.com/adf/faces/rich">
```

```
    <af:document title="webnewsdbfeeds.jsf" id="d1"
binding="#{backingBeanScope.backing_webnewsdbfeeds.d1}">
        <af:form id="f1" binding="#{backingBeanScope.backing_webnewsdbfeeds.f1}">
            <p xmlns="http://www.w3.org/1999/xhtml">
                <af:outputLabel value="Döcu Content WebNews DB Feeds" id="ol1"
binding="#{backingBeanScope.backing_webnewsdbfeeds.ol1}"
                                inlineStyle="font-size:x-large; font-
weight:bolder;"/>
            </p>
            <p xmlns="http://www.w3.org/1999/xhtml">
                <af:button text="XML Data to DB" id="b1"
binding="#{backingBeanScope.backing_webnewsdbfeeds.b1}"
                           action="#{backingBeanScope.backing_
webnewsdbfeeds.b1_action}"/>
            </p>

        </af:form>
    </af:document>
    <!--oracle-jdev-comment:auto-binding-backing-bean-name:backing_webnewsdbfeeds--
>
</f:view>
```

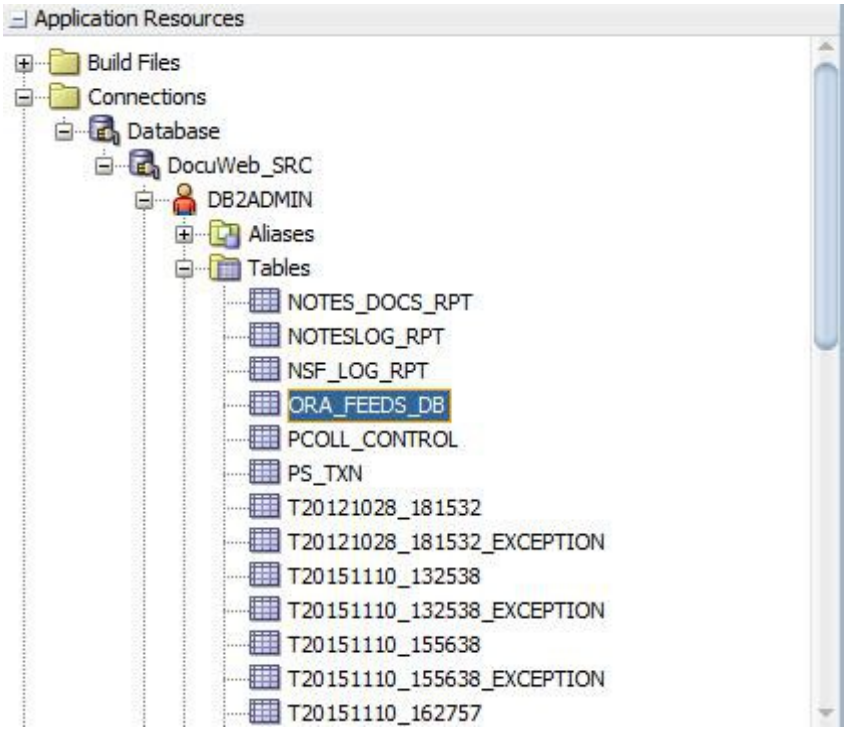
JavaServer Faces(JSF) Submit

Form Design

Added Information, Quick look at the App structure



Quick look at the Database structure



Backing code running JSF File: Copy and paste below Backing Bean in your environment to work with the JSF file submitted above, grab records from XML file in URL and load records into DB2 back-end... Once the JSF page runs and you have accessed the button, the IBM Domino App will release the XML file allowing JDeveloper to collect the data.

Webnewsdbfeeds.java

```
/**
 * Created: 2016.08.14.12.33.PM
 * New Oracle XML Feeds data Posting via JSF form and Backing Bean
 * for Döcu Content App in IBM Notes Domino Designer
 *
 */
package com.dokoll.solutions.inc.DocuWebNews.view.backing; import

com.dokoll.solutions.inc.DocuWebNews.model.DB2Connector;

import javax.xml.parsers.DocumentBuilder; import

javax.xml.parsers.DocumentBuilderFactory;

import oracle.adf.view.rich.component.rich.RichDocument;
import oracle.adf.view.rich.component.rich.RichForm; import
oracle.adf.view.rich.component.rich.nav.RichButton;
import oracle.adf.view.rich.component.rich.output.RichOutputLabel;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

import org.xml.sax.SAXException;
import org.xml.sax.SAXParseException;


//
//...
import java.net.*;
import java.sql.*;
import java.io.*;

import java.io.PrintWriter;
//import javax.xml.parsers.DocumentBuilder;.

//import javax.xml.parsers.DocumentBuilderFactory;

//import org.w3c.dom.Document;
//import org.w3c.dom.Element;
//import org.w3c.dom.Node;
//import org.w3c.dom.NodeList;
//import org.xml.sax.SAXException;
//import org.xml.sax.SAXParseException;

import java.net.URL;
import java.net.HttpURLConnection;
```

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;


import javax.faces.context.FacesContext;


public class Webnewsdbfeeds {
    private RichForm f1;
    private RichDocument d1;
    private RichOutputLabel ol1;
    private RichButton b1;

    public void setF1(RichForm f1) {
        this.f1 = f1;
    }

    public RichForm getF1() {
        return f1;
    }

    public void setD1(RichDocument d1) {
        this.d1 = d1;
    }

    public RichDocument getD1() {
        return d1;
    }

    public void setOl1(RichOutputLabel ol1)
        { this.ol1 = ol1;
    }

    public RichOutputLabel getOl1() {
        return ol1;
    }

    public void setB1(RichButton b1) {
        this.b1 = b1;
    }

    public RichButton getB1() {
        return b1;
    }

    public void b1_action() {

        //TO DO: Send ip and other info to back-end if page
        firing //this button code
            String columnName = "";
            String dataOutput = "";

        // entering try catch
```

```
        try {
            // set up document read/build
            DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
            DocumentBuilder db = dbf.newDocumentBuilder();
            Document doc =
db.parse("http://localhost/docucontent.nsf/javaagentdataforwebnewshttp.xml");

            /  normalize text representation
            doc.getDocumentElement().normalize();
            System.out.println("Root element of the doc is " +
doc.getDocumentElement().getNodeName());

            /  ...
            /  Get count of values per tags

            /  perform count of documents retrieved int
            totalIssues = listOfIssues.getLength();
            System.out.println("Total number of issues/users : " + totalIssues);

            /  loop through issues and fill nodes
            for (int s = 0; s < listOfIssues.getLength(); s++) {

                /  ...
                /  Set up issuesNode to gather data into
                elements Node issuesNode = listOfIssues.item(s);
                if (issuesNode.getNodeType() == Node.ELEMENT_NODE) {

                    // ...
                    Element issuesElement = (Element) issuesNode;

                    /  ...
                    /  load to respective elements
                    NodeList userNameList =
issuesElement.getElementsByTagName("username");
                    Element issuesNameElement = (Element) userNameList.item(0);

                    /  ...
                    /  feed values outbound via each
                    element NodeList textIssuesNameList =
issuesNameElement.getChildNodes();
                    System.out.println("username : " +
(textIssuesNameList.item(0)).getNodeValue().trim());


                    /  ...
                    /  load to respective elements
                    NodeList issuesPageList =
issuesElement.getElementsByTagName("pageid");
                    Element issuesPageElement = (Element) issuesPageList.item(0);

                    /  ...
                    /  feed values outbound via each
                    element NodeList textIssuesPageList =
issuesPageElement.getChildNodes();
                    System.out.println("pageid : " +
(textIssuesPageList.item(0)).getNodeValue().trim());
```

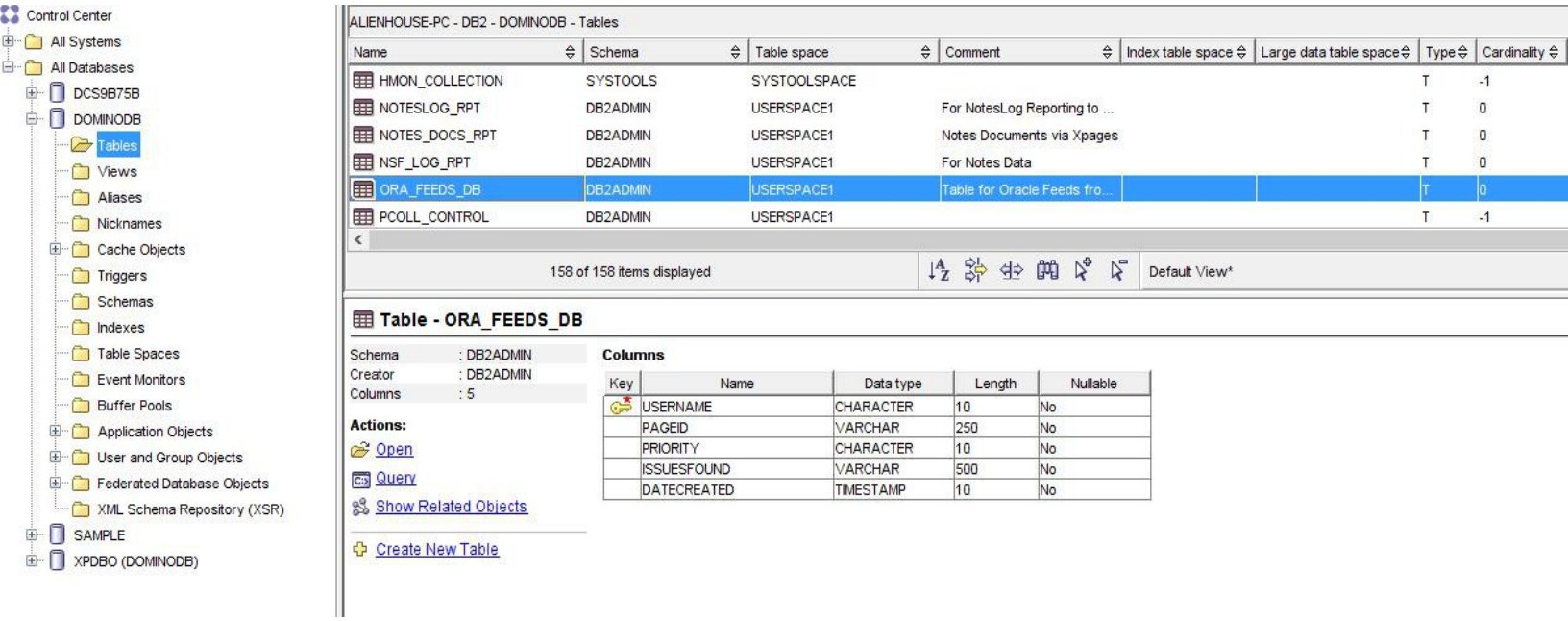
```
                    /   ...
                    /   load to respective elements
                    NodeList userPrioList =
issuesElement.getElementsByTagName("priority");
                    Element issuesPrioElement = (Element) userPrioList.item(0);

                    /   ...
                    /   feed values outbound via each
                    element NodeList textIssuesPrioList =
issuesPrioElement.getChildNodes();
                    System.out.println("priority : " +
(textIssuesPrioList.item(0)).getNodeValue().trim());


                    /   ...
                    /   load to respective elements
                    NodeList userNotesList =
issuesElement.getElementsByTagName("usernotes");
                    Element issuesNotesElement = (Element) userNotesList.item(0);

                    /   ...
                    /   feed values outbound via each
                    element NodeList textIssuesNotesList =
issuesNotesElement.getChildNodes();
                    System.out.println("usernotes : " +
(textIssuesNotesList.item(0)).getNodeValue().trim());


                    /   ...
                    /   load to respective elements
                    NodeList userStampList =
issuesElement.getElementsByTagName("timestamp");
                    Element issuesStampElement = (Element) userStampList.item(0);

                    /   ...
                    /   feed values outbound via each
                    element NodeList textIssuesStampList =
issuesStampElement.getChildNodes();
                    System.out.println("timestamp : " +
(textIssuesStampList.item(0)).getNodeValue().trim());


                    /   load results to ColumnNames bound for CSV
                    file String IssuesName =
((textIssuesNameList.item(0)).getNodeValue().trim());

                    String IssuesPage =
((textIssuesPageList.item(0)).getNodeValue().trim());


                    /   load results to ColumnNames bound for CSV
                    file String IssuesPriority =
((textIssuesPrioList.item(0)).getNodeValue().trim())
                    ; String IssuesUserNotes =
((textIssuesNotesList.item(0)).getNodeValue().trim());
```

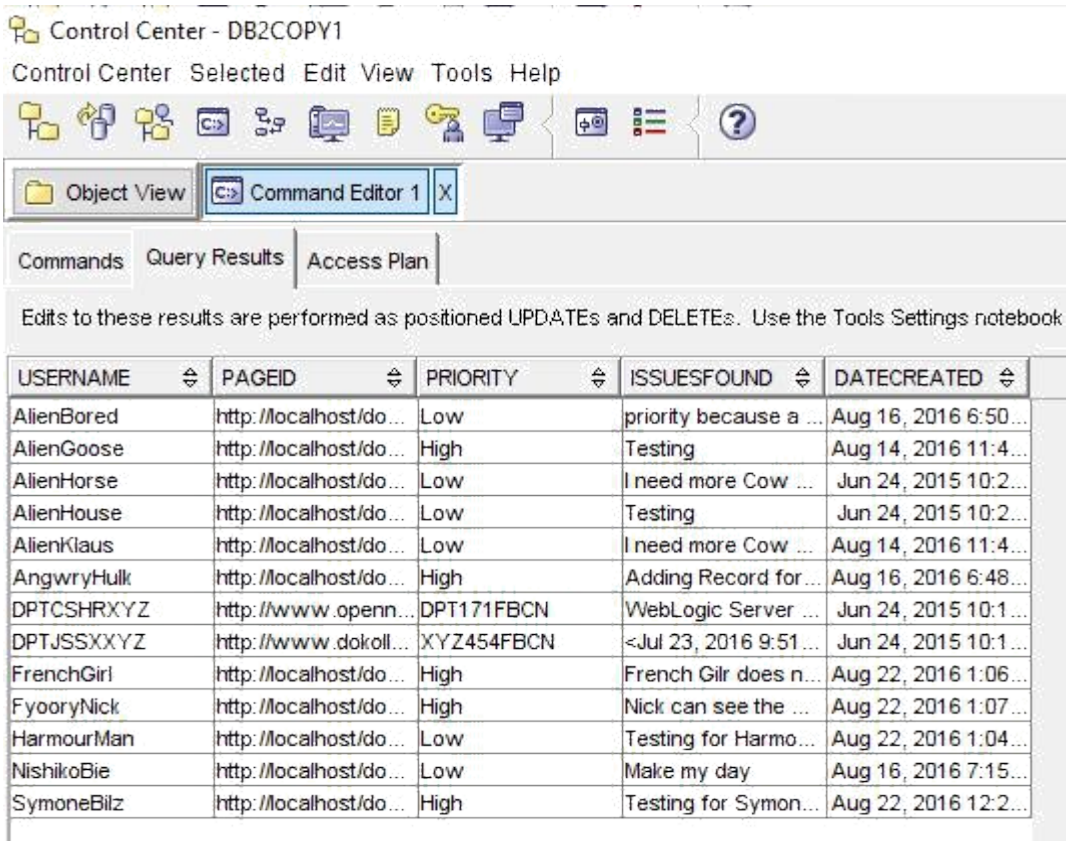
```
                    /   load results to ColumnNames bound for CSV
                    file String IssuesTimeStamp =
((textIssuesStampList.item(0)).getNodeValue().trim());
                    System.out.println("This is for Debugging measures: ");
                    /   pre-arrange column values for CSV
                    file columnName =
                        "username" + "," + "pageid" + "," + "priority" + ","
+ "usernotes" + "," + "timestamp" +
                        "\n"; // ...
                    // load values to columns set up
                    dataOutput =
                        dataOutput + IssuesName + "," + IssuesPage + ","
+ IssuesPriority + "," + IssuesUserNotes +
                        "," + IssuesTimeStamp + "\n";

                    /   add debug messages
                    /   TODO: Remove in Prod
                    System.out.println(columnName + dataOutput);


                        //setup variable(s) to grab values from XML, add to
                        DB2 String strFileName = IssuesName.toString();
                        String strFilePage = IssuesPage.toString(); String
                        strFilePrio = IssuesPriority.toString(); String
                        strFileNotes = IssuesUserNotes.toString(); String
                        strFileStamp = IssuesTimeStamp.toString();
                        //TODO: Prepare a Statrement here to Search DB2 prior
to Submitting records to the DB
                        /   if this is not done, you will see a
                        getNextException //establish a connection
                        Connection connection = DB2Connector.getConnection();

                        System.out.println("Entering Insert
                        query..."); //prepare INSERT
                        PreparedStatement prep =
                            connection.prepareStatement("insert into
DB2ADMIN.ORA_FEEDS_DB values (?,?,?,?,?)");

                        //Added Data for report
                        //2016.07.26.10.01.PM
                        prep.setString(1, strFileName);
                        prep.setString(2, strFilePage);
                        prep.setString(3, strFilePrio);
                        prep.setString(4, strFileNotes);
                        prep.setString(5, strFileStamp);

                        prep.addBatch();

                        System.out.println("Values added...");
                        connection.setAutoCommit(false);
                        prep.executeBatch();

                        connection.setAutoCommit(true);
                        //...
                        connection.close();
                    }
```

```
            }
        } catch (SAXParseException err) {
            System.out.println("** Parsing error" + ", line " + err.getLineNumber()
 + ", uri " + err.getSystemId());   + err.getMessage());
            System.out.println(" "

        } catch (SAXException e) { Exception
            x = e.getException();
            ((x == null) ? e : x).printStackTrace(); }

        catch (Throwable t) {
            t.printStackTrace();
        }

    } // end of program
}
```

DB2 Back-end with XML Data: Nothing fancy here, just the DB2 data submitted during this exercise. In this case, existing data were removed from the IBM Notes Domino App to avoid the getNextException error

IBM DB2 Data





Conclusion:

You are now able to run a JSF App using a BackingBean to submit XML data to DB2 database from a URL located on IBM Notes Domino App. For all Questions and comments, please add a Quick note to our Contact form, or visit our social media networks.

Contact
http://www.dokollsolutionsinc.com/apptrendscontactemail.php

Facebook
https://www.facebook.com/Dököll-Solutions-Inc-233555900032117/

Google+
https://plus.google.com/u/0/+DökköllSolutions/posts

Twitter
https://twitter.com/DokollSolutions

YouTube
https://www.youtube.com/channel/UCSImDTpK0oe7QrPsYOE4nww

Dököll Solutions, Inc.
version: 2016.08.22.10.21.PM