# JDeveloper

## Read Lotus Notes Data via URL Part 2

# Introduction:

Read App Data from Lotus Notes Database into Java Server Faces Page on JDeveloper, running on Weblogic Server

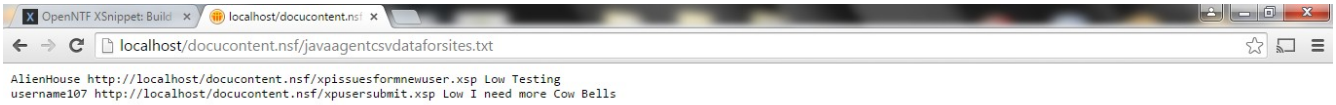
# Use Existing Code

# HTTPCSVDataJavaAgent

```
/**
 * Created: 2011.04.15.11.54.AM
 * LoggedInHTTPUserXMLDataJavaAgent
 * XML data for Xpage to be read by JavaBean
 */

//load imports
import lotus.domino.*;
import lotus.domino.local.Database;

import java.io.PrintWriter;

/**
 * @author Dököll Solutions, Inc.
 * @version 2011.04.15.11.54.AM
 *
 */

// begin class
public class JavaAgent extends AgentBase {

      // open method, this actually runs the whole App
      public void NotesMain() {

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

```
            try {
                  // open our session...
                  Session session = getSession();
                  // load info to console for debugging purposes
                  System.out
                              .println("HTTPCSVDataJavaAgent System User, We've got
a session..."
                                          + session);

                  // load agentContext
                  AgentContext agentContext = session.getAgentContext();
                  // ...
                  System.out.println("got HTTPCSVDataJavaAgent agentContext..."
                              + agentContext);
                  // find database based on session found
                  Database database = (Database) agentContext.getCurrentDatabase();
                  // Find view in question according to current database
                  View view = database.getView("IssuesListings");
                  System.out.println("View Obtained..." + view);

                  // declare document variables
                  Document currDoc;
                  Document tempDoc;
                  // grab first doc
                  currDoc = view.getFirstDocument();

                  // PrintWriter...
                  PrintWriter pw = getAgentOutput();
                  System.out
                              .println("URL:
http://localhost/docucontent.nsf/javaagentcsvdataforsites.txt");

                  // Content type set at Txt
                  // TODO: Figure out a better version to load a CSV file to
browser
                  // currently, the csv downloads to local directory
                  pw.println("Content-type:text/txt");

                  while (currDoc != null) {

                        // write records from view into a Text file and show it in
the
                        // browser
                        pw.println(currDoc.getItemValueString("UserID") + " "
                                    + currDoc.getItemValueString("PageID") + " "
                                    + currDoc.getItemValueString("priority") + " "
                                    + currDoc.getItemValueString("issues"));

                        // Get next document
                        tempDoc = view.getNextDocument(currDoc);
                        // recycle currDoc
                        currDoc.recycle();
                        // set currDoc to tempDoc
                        currDoc = tempDoc;
                  }

            } catch (Exception e) {
                  e.printStackTrace();
            }
```

```
        }
}
```





the data is available here, will grab it from TXT and feed JDeveloper that way

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++>>
2015.06.30.11.19.AM

ok, we're back, let's get the show on the road..  I am currently writing a class right from DDE and will port it to JDeveloper

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++>>
2015.07.04.10.29.AM

**other plans, still the same idea** but rather than additionally putting records into a file or showing stuff to browser, we will throw it into DB2 columns, by combining the following samples

# SAMPLE 1

```
//...
      //Button code
  public static void doGetCSVForDB2Data()
  {

        //...
        //Declare and initialize connection items
        Statement stmt=null;
        ResultSet rsRata=null;
      Connection conn=null;
        //Entering try catch
    try
    {
        // load driver into memory
        Class.forName("org.relique.jdbc.csv.CsvDriver");
        // get a connection
        conn =
DriverManager.getConnection("jdbc:relique:csv:C:\\temp\\CSV_DATA\\");
        //2012.04.21.6.02.PM
        //TO DO: Attempt using Prepared Statement here...
        stmt = conn.createStatement();

        // Select * from BULK_CSV_COM_DATA_SPLIT.csv
        rsRata = stmt.executeQuery("SELECT * FROM BULK_CSV_COM_DATA_SPLIT");
        int docount = 0;
        // loop through resultSety
        while (rsRata.next())
        {
              //TO DO: Write a separate JavaBean to build setters
              //that can used in Xpages to reload entries newly added back to the
App...
            String strFileDate = rsRata.getString("FileDate");
            String strFileTime = rsRata.getString("FileTime");
            String strErrorAmPm = rsRata.getString("ErrorAmPm");
            String strErrorRequest = rsRata.getString("ErrorRequest");
            String strErrorOrigin = rsRata.getString("ErrorOrigin");
            String strErrorDescrip = rsRata.getString("ErrorDescrip");
            String strErrorLang = rsRata.getString("ErrorLang");

            docount++;
            //connect to NSF back-end to load Data in
            Database database= (Database) FacesContext.getCurrentInstance()
            .getApplication().getVariableResolver()
            .resolveVariable(FacesContext.getCurrentInstance(), "database");
            System.out.println("Connected to database..." + database);


        System.out.println("Entering actual connection...");
        System.out.println("Insert: "+rsRata.getString("FileDate")+ " to DB2");
        System.out.println("Insert: "+rsRata.getString("FileTime")+ " to DB2");
        System.out.println("Insert: "+rsRata.getString("ErrorAmPm")+ " to DB2");
```

```
        System.out.println("Insert: "+rsRata.getString("ErrorRequest")+ " to DB2");
        System.out.println("Insert: "+rsRata.getString("ErrorOrigin")+ " to DB2");
        System.out.println("Insert: "+rsRata.getString("ErrorDescrip")+ " to DB2");
        System.out.println("Insert: "+rsRata.getString("ErrorLang")+ " to DB2");
        Connection connection =  DB2Connector.getConnection();
      System.out.println("Entering query...");
        PreparedStatement prep = connection.prepareStatement("insert into
DB2ADMIN.NOTES_DOCS_RPT values (?,?,?,?,?,?,?,?)");

//        //Added Data for report
//        //2012.10.28.9.03.PM
        prep.setString(1, DocDate);
        prep.setString(2, strFileDate);
        prep.setString(3, strFileTime);
        prep.setString(4, strErrorAmPm);
        prep.setString(5, strErrorRequest);
        prep.setString(6, strErrorOrigin);
        prep.setString(7, strErrorDescrip);
        prep.setString(8, strErrorLang);
        prep.addBatch();

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

        //prep.close();
       //connection.close();


      }



      // clean up
      rsRata.close();
      stmt.close();
      conn.close();
    }
    catch(Exception e)
    {
      e.printStackTrace();
    }
  }//end of program...
```

# SAMPLE 2

```
/**
 * Created from copy: 2015.06.30.11.23.AM
 * Item to read XML Data into CSV via URL to Feed JDeveloper JSF page
 */
package com.dokoll.solutions.inc.db2.test;

import java.io.BufferedOutputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.net.URL;
import java.net.URLConnection;

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;

/**
 * @author Dököll Solutions, Inc.
 * @version: 2015.06.30.11.23.AM
 *
 */
public class GetXMLURLDataIntoCSVBean {


      // run program/button code
      public void getXMLKeywords() {
        // Set up column and data variables
        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/dashboardcha
rtdata.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
          NodeList listOfSites = doc.getElementsByTagName("person");
```

```
// perform count of documents retrieved
int totalSites = listOfSites.getLength();
System.out.println("Total number of sites/customers : " + totalSites);

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

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

    //...
    Element siteElement = (Element) siteNode;

    //...
    //load to respective elements
    NodeList siteNameList = siteElement
        .getElementsByTagName("sitename");
    Element siteNameElement = (Element) siteNameList.item(0);

    //...
    //feed values outbound via each element
    NodeList textSiteNameList = siteNameElement.getChildNodes();
    System.out
        .println("SiteName : "
            + ((Node) textSiteNameList.item(0))
                .getNodeValue().trim());
    //...
    //load to respective elements
    NodeList siteNumberList = siteElement
        .getElementsByTagName("sitenumber");
    Element siteNumberElement = (Element) siteNumberList.item(0);

    //...
    //feed values outbound via each element
    NodeList textSiteNumberList = siteNumberElement.getChildNodes();
    System.out
        .println("SiteNumber : "
            + ((Node) textSiteNumberList.item(0))
                .getNodeValue().trim());

    // load results to ColumnNames bound for CSV file
    String SiteName = (((Node) textSiteNameList.item(0))
        .getNodeValue().trim());
    String SiteNumber = (((Node) textSiteNumberList.item(0))
        .getNodeValue().trim());
    // pre-arrange column values for CSV file
    columnName = "SiteName" + "," + "SiteNumber" + "\n"; // ...
    // load values to columns set up
    dataOutput = dataOutput + SiteName + "," + SiteNumber
        + "\n";
    // add debug messages
    System.out.println("This is for Debugging measures: "
        + columnName + dataOutput);
    // ...
    // Instantiate file location...
    DataOutputStream outBound = new DataOutputStream(
        new BufferedOutputStream(
```

```
                    new FileOutputStream(
                        "c:/temp/CSV_DATA/UserNewLineOutboundInformation.csv")));
                // write data into columns set up
                outBound.writeBytes(columnName + dataOutput);
                // ...
                //close the writer
                outBound.close();

              }

            }

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

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

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

        }// end of program

}
```

version: 2015.07.05.12.05.AM