JavaAgent/Backing Bean

URL connect, XML on C:/

URL2XML

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, create an XML document using a JavaAgent, perform a sort of handshake with the XML data, pulled into a bean class via URL. Build a new XML file with URL data, submit file to your C drive.

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 XML file to URL
At this point we assume Domino Designer 8.5.2 is downloaded/installed and a new JavaAgent, is already created. You are building XML tags with data from a view in your database. You will pull these tags through URL and save as a file to local C drive, or drive of choice. Copy and paste below code to your environment for each code samples.

TIP: Add an alias name via the JavaAgent's properties, serves the purpose of the name of the XML file: **`dashboardchartdata.xml`**

JavaAgent.java

[CODE]

```
/**
 * Created: 2011.12.27.3.50.PM
 * Program: DashboardChartJavaAgent (JavaAgent.java)
 * XML data for Xpages/Charts
 */

//load imports
import lotus.domino.*;
import java.io.PrintWriter;
```

```
/**
 * @author Dököll Solutions, Inc.
 * @version 2011.12.27.3.50.PM
 *
 */


//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, grab errors near the end
		try {
			//open our session...
			Session session = getSession();
			//load info console for debugging purposes
			System.out.println("We've got a session..." + session);
			//load agentContext, grab database in question
			AgentContext agentContext = session.getAgentContext();
			Database currdb = agentContext.getCurrentDatabase();

			//load info console for debugging purposes
			System.out.println("connected to database..." + currdb);
			//load view in question
			View view = currdb.getView("SiteList");
			//load info console for debugging purposes
			System.out.println("We've got our view!" + view);
			//load first document
			Document doc;
			Document tempDoc;
			doc = view.getFirstDocument();

			//...
			PrintWriter pw = getAgentOutput();
			//...
			pw.println("Content-type:text/xml");
			pw.println("");
			//Write out the Opening Tags
			pw.println("<?xml version='1.0' encoding='UTF-8' ?>");
			pw.println("<PieDataset>");
			//loop though docs and fill xml tags
			while (doc != null) {

				//...
				pw.println("<item>");

				pw.println("<key>");
				pw.println(doc.getItemValueString("SiteName"));
				pw.println("</key>");

				pw.println("<value>");
				pw.println(doc.getItemValueString("SiteNumber"));
				pw.println("</value>");

				pw.println("</item>");
```

```
                              tempDoc = view.getNextDocument(doc);
                              //recycle, free up memory...
                              doc.recycle();
                              //...
                              doc = audoc;
                        }
                  //close the root tag
                  pw.println("</PieDataset>");

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

Bean it!

Create a bean, you are now ready to pull in XML data from URL, save it to your C drive; data that can be loaded to Xpages, perhaps styled with an XSL sheet, and load to browser. You can use `XMLURL2FileCreate.GetXMLFromURL()` in your Xpage button to run the code (or use SSJS onBeforePageLoad). Copy and paste below to your environment, areas of interest have been highlighted for your convenience.

XMLURL2FileCreate.java

```
/**
 * Created: 2012.04.08.3.50.AM
 * Program: XMLURL2FileCreate.java
 * XML data for Xpages/Charts
 */



package com.dokoll.solutions.inc.xml.parsers;

//load imports
import java.net.*;
import java.io.*;
import javax.faces.context.FacesContext;
import lotus.domino.local.Database;

/**
 * @author Dököll Solutions, Inc.
 * @version 2012.04.08.3.50.AM
 *
 */



public class XMLURL2FileCreate {


      public void GetXMLFromURL() {
            try {
```

```
                    //TO DO: Send ip and other info to back-end if page firing
                    //via button code by multiple users...

                    //Get XML file from URL.

                    URL xmlURL = new URL("http", "localhost", 80,
                                "/filebin.nsf/dashboardchartdata.xml");
                    // Set up Connection to URL, grab file
                    URLConnection connectXML = xmlURL.openConnection();
                    // connect to URL.... grab file
                    connectXML.connect();

                    // Build URL file into new XML file
                    // ...
                    PrintWriter prWriter = new PrintWriter(new FileWriter(
                                "c:\\temp\\XML_DATA\\documentfromurl.xml"));
                    // Read URL data into new XML file
                    BufferedReader bufferRead = new BufferedReader(
                                new InputStreamReader(connectXML.getInputStream()));
                    // ...
                    String dataRead = bufferRead.readLine();

                    // ...
                    while (dataRead != null) {
                          prWriter.println(dataRead);
                          dataRead = bufferRead.readLine();

                    }

                    //close writer...
                    prWriter.close();

              } catch (Exception e) {
                    System.out.println("Error: " + e);

              }

        }

}
```

Conclusion:
You can now put data where you need it in XML file format to be used by other software/systems.

TIP: Use the XML file on C:/ as part of Xpages via iFrames, load to reusable custom controls, styled with XSL stylesheets, or build and embed charts using the XML file as DataSource.

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

Thank you for coming...