

Database Viewer using Web Dynpro Java

Applies to:

SAP NetWeaver 04s.

Summary

This article will explain in step by step about how to read the database contents in Web Dynpro and also about dynamic creation of view elements and context attributes.

Author(s): Giridharan S

Company: SAP Labs India

Created on: 21 January 2007

Author Bio



Giridharan S is working as a Developer in NetWeaver Solution and Platform Management team, **SAP Labs India**

Table of Contents

Introduction	3
Prerequisites	3
Step by Step Solution for Building Database Viewer:	4
1. Create Web Dynpro Project	4
2. Create Web Dynpro Application, Component, Window	4
3. Create Contexts in Component Controller	4
3.1 Create Method for Retrieving Database Table Contents in Component Controller	4
3.2 Create Views and Plugs.....	7
4. View Implementation.....	7
4.1 Input View Implementation.....	7
4.2 Result View Implementation	8
4.3 Deploy and Run the Application.....	10
Related Content.....	11
Copyright.....	12

Introduction

This article explains in step by step about how to read the database contents in Web Dynpro for java and also about dynamic creation of view elements and context attributes for the corresponding database table. By following these steps, you can have your own database viewer which will be similar to ABAP transaction SE16, i.e Underlying database is abstracted to the UI which is achieved through JDBC standards.

Code and view layouts mentioned in this article could be optimized based on your need.

Prerequisites

Access to SAP NetWeaver AS Java and SAP NetWeaver Developer Studio installed in your system

Step by Step Solution for Building Database Viewer:

1. Create Web Dynpro Project

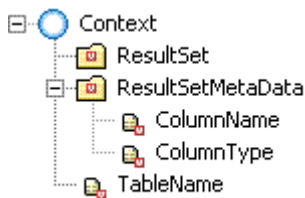
Open SAP NetWeaver Developer Studio. Click on New-> Web Dynpro Project. Name the project as "DBViewer".

2. Create Web Dynpro Application, Component, Window

Open the created project. Right Click on Application and click on "Create Application". Name the application as "DBViewer" and specify the package details. Click on Next. Choose the option "Create a new Web Dynpro component" . Deselect the option of creating new view and accept default values.

3. Create Contexts in Component Controller

Open the Web Dynpro Component Controller DBViewer. Open the Context Tab. Create node named "ResultSet" and another node named "ResultSetMetaData". Create two attributes named "ColumnName" and "ColumnType" under "ResultSetMetaData" node. Create attribute named "TableName" under Root context as in the following figure.



3.1 Create Method for Retrieving Database Table Contents in Component Controller

Open the Methods tab of Component Controller and create a method named "RetrieveData".

Open the implementation of this method and paste the following code.

```

IWDMessageManager messageMgr = wdComponentAPI.getMessageManager();
wdContext.nodeResultSet().invalidate();
wdContext.nodeResultSetMetaData().invalidate();

// find if ResultSet has dynamic attributes
boolean hasAttr = false;
for (Iterator iter =
wdContext.nodeResultSet().getNodeInfo().iterateAttributes();
iter.hasNext();) {
    hasAttr= true;
    break;
}
// if ResultSet has some dynamic attributes, remove them first

```

```

    if(hasAttr == true)
    {
        wdContext.nodeResultSet().getContext().reset();
    }

    try {

        // Get the database connection
        InitialContext dbInitContext = new InitialContext();
        Properties sysProperties = System.getProperties();
        String sysname = sysProperties.getProperty("SAPSYSTEMNAME");
        String dbName = "jdbc/" + "SAP" + sysname + "DB";
        DataSource dataSource = (DataSource) dbInitContext.lookup(dbName);
        Connection conn = dataSource.getConnection();
        String tableName = wdContext.currentContextElement().getTableName();
        String SelectStmt = "Select * from" .concat(" " + tableName);
        // Prepare the SQL statements
        PreparedStatement stmt = conn.prepareStatement(SelectStmt);
        // Execute the Query
        ResultSet resultSet = stmt.executeQuery();
        // Get the result set metadata
        ResultSetMetaData resultMetaData = resultSet.getMetaData();
        // resultMetaData contains DB Column details

        int numColumns = resultMetaData.getColumnCount();

        // Create an element of ResultSetMetadata for every table column
        (attribute)
        // Create a dynamic attribute for the context of ResultSet Node for
        every table column
        String[] attributes = new String[numColumns];
        for (int columnIndex = 1; columnIndex <= numColumns; columnIndex++) {
            IResultSetMetaDataElement element =
wdContext.nodeResultSetMetaData().createResultSetMetaDataElement();
            String columnName = resultMetaData.getColumnName(columnIndex);
            String columnType = resultMetaData.getColumnTypeName(columnIndex);
            element.setColumnName(columnName);
            attributes[columnIndex-1] = columnName;
        }
    }
}

```

```

        element.setColumnTypes(columnTypes);
        // Adding dynamic attribute to ResultSet Node

wdContext.nodeResultSet().getNodeInfo().addAttribute(columnName, "java.lang.S
tring");

        wdContext.nodeResultSetMetaData().addElement(element);
    }

    // Loop through every record of DB Table and create an element in
    ResultSet Node
    while(resultSet.next())
    {
        IResultSetElement element =
wdContext.nodeResultSet().createResultSetElement();
        for (int recordIndex = 0; recordIndex < attributes.length;
recordIndex++) {
            // If the Object is null, make it as empty
            if(resultSet.getObject(attributes[recordIndex]) != null)
            {

element.setAttributeValue(attributes[i],resultSet.getObject(attributes[reco
rdIndex]).toString());
                } else
                {

element.setAttributeValue(attributes[recordIndex], "");
                }
            }
            wdContext.nodeResultSet().addElement(element);
        }

        stmt.close();
        conn.close();
    } catch (NamingException e) {
        messageMgr.reportException(e.getMessage(), false);
    } catch (SQLException e) {
        messageMgr.reportException(e.getMessage(), false);
    }

```

```

    } catch (Exception e)
    {
        messageMgr.reportException(e.getMessage(), false);
    }

```

3.2 Create Views and Plugs.

Create a view named “TableInputView” (for getting the table name from the user) and “TableResultView” (for displaying the database records). Map TableName Attribute from Component Controller to TableInputView. Map ResultSet and ResultSetMetaData Nodes from Component Controller to TableResultView.

Create layout for both the views as in the following picture.

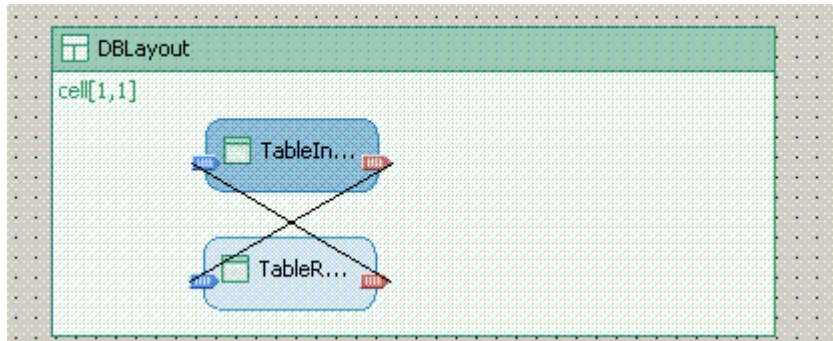
(TableInputView)



(TableResultView)



Open the “DBViewer” window and create a Grid Layout (1 Column by 1 Row). Embed these two views inside the layout and make “TableInput” view as default. Create inbound and outbound plugs in both the views and connect them through data link (as in the following picture)



4. View Implementation

4.1 Input View Implementation

Open the View “TableInputView” and create an action for “ShowRecords” and paste the following code for the action implementation. This action will be triggered on click of “Show Records” Button.

```

wdThis.wdGetDBViewerController().RetrieveData();
wdThis.wdFirePlugToResFromInput();

```

4.2 Result View Implementation

Open the View "TableResultView", create a context node called ViewResultSet (Bind this context as datasource in the table) and paste the following code in wdDoModifyView.

```
// First, Delete all the dynamically created view elements
view.resetView();

// Get the Root Transparent Container
IWDTransparentContainer container = (IWDTransparentContainer)
view.getElement("RootUIElementContainer");

// Get the Table
IWDTTable table = (IWDTTable) view.getElement("Table");

        for (int columnIndex = 0; columnIndex <
wdContext.nodeResultSetMetaData().size(); columnIndex++) {

            // Create Table Cell Editor
            IWDTTextView editor = (IWDTTextView)
view.createElement(IWDTTextView.class,
"TableColumnTextViewEditor"+columnIndex);

            editor.bindText(wdContext.nodeViewResult().getNodeInfo().getAttribute(wdCo
ntext.nodeResultSetMetaData().getResultSetMetaDataElementAt(columnIndex).get
ColumnName()));

            // Create Table Cell Header

            String columnHeader1 = "TableColumnCaption"+columnIndex;
            IWDCaption columnCaption1 = (IWDCaption)
view.createElement(IWDCaption.class, columnHeader1);

            columnCaption1.setText(wdContext.nodeResultSetMetaData().getResultSetMetaD
ataElementAt(columnIndex).getColumnName());

            // Create Table Column
            IWDTTableColumn column = (IWDTTableColumn)
view.createElement(IWDTTableColumn.class, "TableColumn" + columnIndex);
            column.setTableCellEditor(editor);
            column.setHeader(columnCaption1);
            table.addGroupedColumn(column);

        }
}
```


And paste the following code in `onPlugFromInputToRes`.

```
// First remove the dynamic attributes of ViewResult context node
    boolean hasAttr = false;
    for (Iterator iter =
wdContext.nodeViewResult().getNodeInfo().iterateAttributes();
iter.hasNext();) {
        hasAttr = true;
        break;
    }
    if(hasAttr == true)
    {
        wdContext.nodeViewResult().getContext().reset();
    }

    // Create dynamic attribute in ViewResult context node
    String[] attributes = new
String[wdContext.nodeResultSetMetaData().size()];
    for (int columnIndex = 0; columnIndex <
wdContext.nodeResultSetMetaData().size(); columnIndex++) {
        String columnName =
wdContext.nodeResultSetMetaData().currentResultSetMetaDataElement().getColumn
nName();
        attributes[columnIndex] = columnName;

        wdContext.nodeViewResult().getNodeInfo().addAttribute(columnName,wdContext
.nodeResultSetMetaData().getNodeInfo().getAttribute("ColumnName").getDataTyp
e());
        wdContext.nodeResultSetMetaData().moveNext();
    }

    // Retrieve the database contents from the controller context node of
ResultSet and assign those values to ViewResult
    for (int recordIndex = 0; recordIndex <
wdThis.wdGetDBViewerController().wdGetContext().nodeResultSet().size();
recordIndex++) {
        IPublicDBViewer.IResultSetElement controllerElement =
wdThis.wdGetDBViewerController().wdGetContext().nodeResultSet().getResultSet
ElementAt(recordIndex);
        IPrivateTableResultView.IViewResultElement viewElement =
wdContext.nodeViewResult().createViewResultElement();
        for (int j = 0; j < attributes.length; j++) {
```

```

        String attrValue =
controllerElement.getAttributeValue(attributes[j]).toString();
        viewElement.setAttributeValue(attributes[j],attrValue);

    }
    wdContext.nodeViewResult().addElement(viewElement);
}

```

Implement the Action for Back Button with the following Code:

```
wdThis.wdFirePlugToInputFromRes();
```

4.3 Deploy and Run the Application

Build the project and deploy the archive in SAP AS Java. Run the application. Application initial page will look like

Enter Table Name:

On click of "Show Records" button page will look like

RESTYPE
EIN
XML
XSD
XSL

Row 1 of 4

If there are multiple columns, all the columns and its corresponding row values will get displayed here.

Related Content

- https://www.sdn.sap.com/irj/sdn/developerareas/Web_Dynpro
- https://www.sdn.sap.com/irj/sdn/developerareas/Web_Dynpro?rid=/library/uuid/49f2ea90-0201-0010-ce8e-de18b94aee2d
- http://help.sap.com/saphelp_erp2005vp/helpdata/en/15/0d4f21c17c8044af4868130e9fea07/frameset.htm

Copyright

© Copyright 2007 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

Any software coding and/or code lines/strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.