Creating Hierarchies using Graphics (Jnet/Network) Instead of Tree in a Table

Applies to:

Composition Environment (CE) Webdynpro for Java. For more information, visit the User Interface Technology homepage.

Summary

Webdynpro Provides a UI Element Called Network which can be used to create graphics. This is rendered at the client side as an applet. ACF is the middle layer which felicitates communication between the applet and the NetworkUI element in terms of events. This component can be used to render Graphs based on Nodes and Edges. This article helps you to explore these features.

Author: Ayyapparaj KV

Company: Bristlecone India Pvt Ltd

Created on: 15 September 2008

Author Bio

Ayyapparaj KV Is a Netweaver Java certified consultant working for Bristlecone

Table of Contents

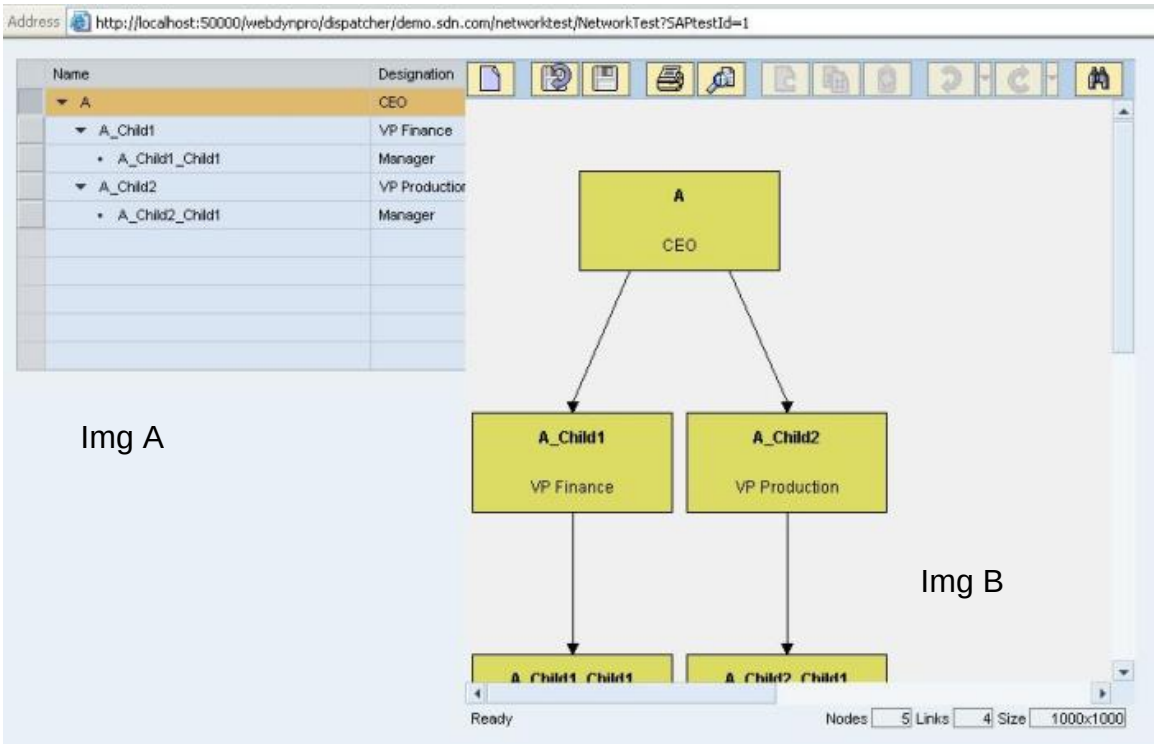
Choices available for creating Hierarchies in Webdynpro

When ever a tree structure is needed in a project, we have basically three choices now.

- Tree: is used to display hierarchically organized data.
- TreeByNesting
- Network: Is a generic editor for network graphics. It can be used to display anything that can be visualized as a set of nodes and links between nodes.

Comparison of a screen rendered as TreeByNesting and Network

Following image depicts the difference in rendering same data using two different UI elements



Note: In the above Image (img A) is created using TreeByNesting and (img B) using Network.

Steps to Create the above Screen

- Create a context Structure based on a recursive node.
- Create a XML file which is based on the JNet Schema.
- Set this XML to a context structure of type Resource.
- Use the Network Ui Element
- Bind the data source property of Network Ui to a context structure of Type Resource.

Context structure used in this example

Creating the XML File

Creation of XML file in this example is based on DOM (Document Object Model).

DOM is an official recommendation of the World Wide Web Consortium (W3C). It defines an interface that enables programs to access and update the style, structure, and contents of XML documents. XMLparsers that support the DOM implement that interface.

Coding

Creating Hierarchy using recursive node

```
public void wdDoInit()
  {
    //@@begin wdDoInit()
     //Hierarchy Creation Code
          //Root
          IEmployeeElement element = wdContext.nodeEmployee()
                       .createAndAddEmployeeElement();
          element.setName("A");
          element.setDesignation("CEO");
          element.setIsChild(false);
          element.setExpanded(true);

    //Child1 of Root
          IEmployeeElement elementChild1 = element.nodeDirectReporting()
                       .createAndAddEmployeeElement();
          elementChild1.setName("A_Child1");
          elementChild1.setDesignation("VP Finance");
          elementChild1.setIsChild(false);
          elementChild1.setExpanded(true);

    //Child2 of Root
          IEmployeeElement elementChild2 = element.nodeDirectReporting()
                       .createAndAddEmployeeElement();
          elementChild2.setName("A_Child2");
          elementChild2.setDesignation("VP Production");
          elementChild2.setIsChild(false);
          elementChild2.setExpanded(true);

    //Child1 of Child1
          IEmployeeElement elementChild1Child1 = elementChild1
                       .nodeDirectReporting().createAndAddEmployeeElement();
          elementChild1Child1.setName("A_Child1_Child1");
          elementChild1Child1.setDesignation("Manager");
          elementChild1Child1.setIsChild(true);

    //Child2 of Child1
          IEmployeeElement elementChild2Child1 = elementChild2
                       .nodeDirectReporting().createAndAddEmployeeElement();
          elementChild2Child1.setName("A_Child2_Child1");
          elementChild2Child1.setDesignation("Manager");
          elementChild2Child1.setIsChild(true);

    // Selecting the first element
          wdContext.nodeEmployee().setTreeSelection(
                       wdContext.nodeEmployee().getElementAt(0));

    //DOM is used in Example for XML Creation
          try {
                factory = DocumentBuilderFactoryImpl.newInstance();
                builder = factory.newDocumentBuilder();
                impl = builder.getDOMImplementation();
```

```
        } catch (Exception e) {
              // TODO: handle exception
        }
        try {
              // Creating a resource based on the fileStream
              byte xmldata[] = createHierarchyForNetwork();
              ByteArrayInputStream is = new ByteArrayInputStream(xmldata);
              IWDResource resource = null;
              ISourceElement xmlElement = wdContext.currentSourceElement();

              resource = WDResourceFactory.createResource(is, "",
                          WDWebResourceType.XML, false);
              // setting xml to the context attribute "Xml" of type resource
              xmlElement.setXml(resource);
        } catch (Exception e) {
              // TODO Auto-generated catch block
              wdComponentAPI.getMessageManager().reportException(e);
        }

    //@@end
  }
```

Creating XML using DOM based on the JNet Schema

Place the following codes at the end, between begin other and end section of your webdynpro controller.

```
//@@begin others
  DocumentBuilderFactory factory = null;
  DocumentBuilder builder = null;
  DOMImplementation impl = null;
  Document document = null;

  //Creating the XML needed for JNet
  public byte[] createHierarchyForNetwork() throws Exception
  {
     //create a document
     document = impl.createDocument(null,"SAPJNetData", null);

     //create root node for JNet
     Element root = document.getDocumentElement();
     root.setAttribute("version", "1.0");

     //Create Type Repository
     Element typeRepository = document.createElement("TypeRepository");
     typeRepository.setAttribute("id", "connections");
     typeRepository.setAttribute("version", "1.0");
     root.appendChild(typeRepository);

     //Create Layout
     Element layout = document.createElement("LAYOUT");
     typeRepository.appendChild(layout);

     //Create Custom Layout
     Element customLayoutType = document.createElement("type");
     customLayoutType.setAttribute("name", "CustomLayout");
     layout.appendChild(customLayoutType);
```

```
Element element = document.createElement("component");
element.setAttribute("index", "0");
element.setAttribute("width", "100%");
element.setAttribute("height", "50%");
customLayoutType.appendChild(element);

element = document.createElement("component");
element.setAttribute("index", "1");
element.setAttribute("width", "100%");
element.setAttribute("height", "50%");
customLayoutType.appendChild(element);

//Create a Label
Element label = document.createElement("LABEL");
typeRepository.appendChild(label);

Element customLabelType = document.createElement("type");
customLabelType.setAttribute("name", "CustomLabel");
customLabelType.setAttribute("inherits", "NodeLabel");
label.appendChild(customLabelType);

element = document.createElement("color");
element.setAttribute("type", "white");
customLabelType.appendChild(element);

element = document.createElement("fillcolor");
element.setAttribute("type", "Green");
customLabelType.appendChild(element);

Element fontElement = document.createElement("font");
customLabelType.appendChild(fontElement);

element = document.createElement("style");
Text style = document.createTextNode("BOLD");
element.appendChild(style);
fontElement.appendChild(element);

//Create Node
Element node = document.createElement("NODE");
typeRepository.appendChild(node);

Element customNodeType = document.createElement("type");
customNodeType.setAttribute("name", "CustomNode");
node.appendChild(customNodeType);

element = document.createElement("layout");
element.setAttribute("type", "CustomLayout");
customNodeType.appendChild(element);

element = document.createElement("label");
element.setAttribute("index", "0");
element.setAttribute("type", "CustomLabel");
customNodeType.appendChild(element);

element = document.createElement("label");
```

```
element.setAttribute("index", "1");
element.setAttribute("type", "NodeLabel");
customNodeType.appendChild(element);

element = document.createElement("plugs");
element.setAttribute("min", "0");
element.setAttribute("position", "SOUTH");
customNodeType.appendChild(element);

element = document.createElement("sockets");
element.setAttribute("min", "0");
element.setAttribute("position", "NORTH");
customNodeType.appendChild(element);

Element customShape = document.createElement("shape");
customNodeType.appendChild(customShape);

element = document.createElement("border");
element.setAttribute("type", "EMPTY");
customShape.appendChild(element);

element = document.createElement("size");
Text size = document.createTextNode("150,75");
element.appendChild(size);
customShape.appendChild(element);

element = document.createElement("filled");
Text filled = document.createTextNode("TRUE");
element.appendChild(filled);
customShape.appendChild(element);

Element fillColor = document.createElement("fillColor");
customShape.appendChild(fillColor);

element = document.createElement("rgb");
Text rgb = document.createTextNode("220,220,100");
element.appendChild(rgb);
fillColor.appendChild(element);

//Create Graph
Element graph = document.createElement("Graph");
graph.setAttribute("version", "1.0");
root.appendChild(graph);

//Layout Direction and View
Element layouts = document.createElement("layouts");
layouts.setAttribute("onLoad", "TREE");
graph.appendChild(layouts);

Element types = document.createElement("types");
Text treeType = document.createTextNode("TREE");
types.appendChild(treeType);
layouts.appendChild(types);

element = document.createElement("direction");
Text direction = document.createTextNode("TOP_BOTTOM");
```

```
    element.appendChild(direction);
    graph.appendChild(element);

    Element view = document.createElement("view");
    view.setAttribute("coordinates", "GRID");
    graph.appendChild(view);

    element = document.createElement("grid");
    Text grid = document.createTextNode("80, 180");
    element.appendChild(grid);
    view.appendChild(element);

    element = document.createElement("offset");
    Text offset = document.createTextNode("40, 0");
    element.appendChild(offset);
    view.appendChild(element);

    getNodes(wdContext.nodeEmployee(), null);

    Source source = new DOMSource(document);
TransformerFactory xformFactory =
               TransformerFactoryImpl.newInstance();
    Transformer idTransform = xformFactory.newTransformer();
    ByteArrayOutputStream sw = new ByteArrayOutputStream();
    Result result = new StreamResult(sw);
    idTransform.transform(source, result);

    return sw.toByteArray();
  }

   public void getNodes(IEmployeeNode node, IEmployeeElement root)
   {
          if(node !=null)
          {
                for(int x=0;x<node.size();x++)
                {
                      IEmployeeElement employeeElement =
                                    node.getEmployeeElementAt(x);
                      Element nodeElement = document.createElement("node");
                      nodeElement.setAttribute("type", "CustomNode");
                      nodeElement.setAttribute("id",
                                  String.valueOf(employeeElement.hashCode()));
                      Element graph =
                                  (Element)document.getDocumentElement().
getElementsByTagName("Graph").item(0);
                      graph.appendChild(nodeElement);

                      Element element = document.createElement("label");
                      element.setAttribute("index", "0");
                      Text label0 =
                      document.createCDATASection(employeeElement.getName());
                      element.appendChild(label0);
                      nodeElement.appendChild(element);

                      element = document.createElement("label");
                      element.setAttribute("index", "1");
```

```
                        Text label1 =
    document.createCDATASection(employeeElement.getDesignation());
                        element.appendChild(label1);
                        nodeElement.appendChild(element);

                        if(root != null)
                        {
                               Element edge = document.createElement("edge");
                               edge.setAttribute("type", "FixLink");
                               graph.appendChild(edge);

                               element = document.createElement("from");
                               element.setAttribute("node",
                                            String.valueOf(root.hashCode()));
                               edge.appendChild(element);

                               element = document.createElement("to");
                               element.setAttribute("node",
                                      String.valueOf(employeeElement.hashCode()));
                               edge.appendChild(element);
                        }
                        getNodes(node.nodeDirectReporting(x), employeeElement);

                }
        }
     }

    //@@end
```

The complete generated XML code

```
<?xml version="1.0" encoding="utf-8"?>
<SAPJNetData version="1.0">
   <TypeRepository id="connections" version="1.0">
          <LAYOUT>
                 <type name="CustomLayout">
                        <component height="50%" index="0" width="100%" />
                        <component height="50%" index="1" width="100%" />
                 </type>
          </LAYOUT>
          <LABEL>
                 <type inherits="NodeLabel" name="CustomLabel">
                        <color type="white" />
                        <fillcolor type="Green" />
                        <font>
                               <style>BOLD</style>
                        </font>
                 </type>
          </LABEL>
          <NODE>
                 <type name="CustomNode">
                        <layout type="CustomLayout" />
                        <label index="0" type="CustomLabel" />
                        <label index="1" type="NodeLabel" />
                        <plugs min="0" position="SOUTH" />
                        <sockets min="0" position="NORTH" />
                        <shape>
                               <border type="EMPTY" />
                               <size>150,75</size>
                               <filled>TRUE</filled>
                               <fillColor>
                                      <rgb>220,220,100</rgb>
                               </fillColor>
                        </shape>
                 </type>
          </NODE>
   </TypeRepository>
   <Graph version="1.0">
          <layouts onLoad="TREE">
                 <types>TREE</types>
          </layouts>
          <direction>TOP_BOTTOM</direction>
          <view coordinates="GRID">
                 <grid>80, 180</grid>
                 <offset>40, 0</offset>
          </view>
          <node id="18353954" type="CustomNode">
                 <label index="0">A</label>
                 <label index="1">CEO</label>
          </node>
          <node id="9014271" type="CustomNode">
                 <label index="0">A_Child1</label>
                 <label index="1">VP Finance</label>
          </node>
          <edge type="FixLink">
```

```
                <from node="18353954" />
                <to node="9014271" />
            </edge>
            <node id="4604630" type="CustomNode">
                <label index="0">A_Child1_Child1</label>
                <label index="1">Manager</label>
            </node>
            <edge type="FixLink">
                <from node="9014271" />
                <to node="4604630" />
            </edge>
            <node id="13499405" type="CustomNode">
                <label index="0">A_Child2</label>
                <label index="1">VP Production</label>
            </node>
            <edge type="FixLink">
                <from node="18353954" />
                <to node="13499405" />
            </edge>
            <node id="22990966" type="CustomNode">
                <label index="0">A_Child2_Child1</label>
                <label index="1">Manager</label>
            </node>
            <edge type="FixLink">
                <from node="13499405" />
                <to node="22990966" />
            </edge>
    </Graph>
</SAPJNetData>
```

Development Component for download

You can download the entire project from here

Related Content

JNet Home Page

JNet XSD

For more information, visit the User Interface Technology homepage. .

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.