ADF Code Corner

021. How-to initially expand a defined set of tree nodes in an ADF bound ADF Faces tree

twitter.com/adfcodecorner

Abstract:

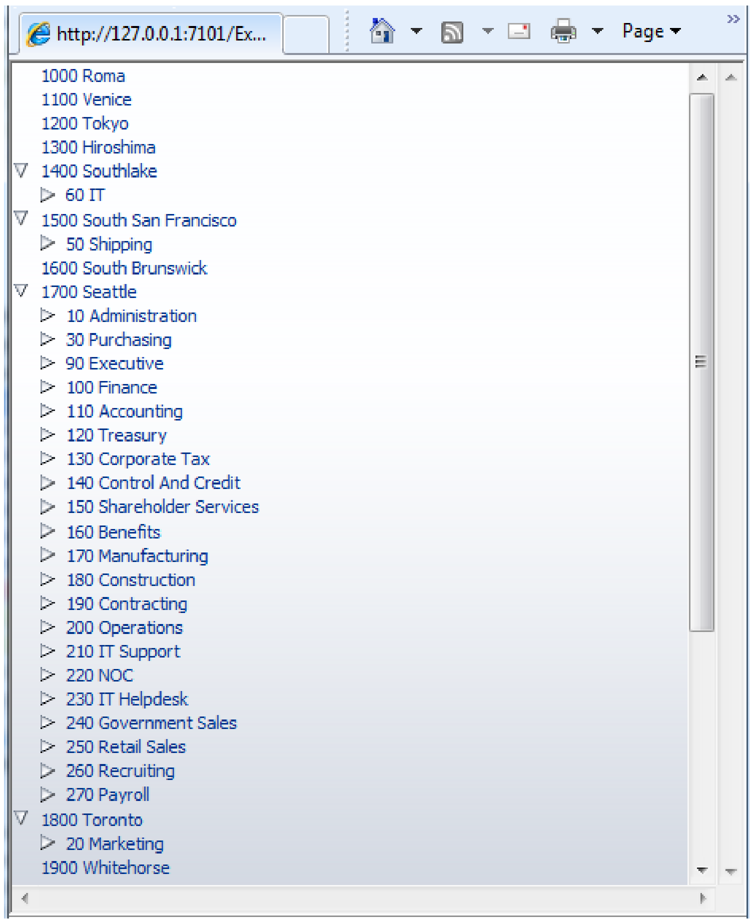
ADF Faces trees usually render with their nodes in discosed state. Some use cases however require the tree to render with a defined set of nodes to be initially expanded.

This how-to article explains how to expand the first level nodes of an ADF bound tree upon initial page rendering.

Author: Frank Nimphius, Oracle Corporation
twitter.com/fnimphiu
23-AUG-2010

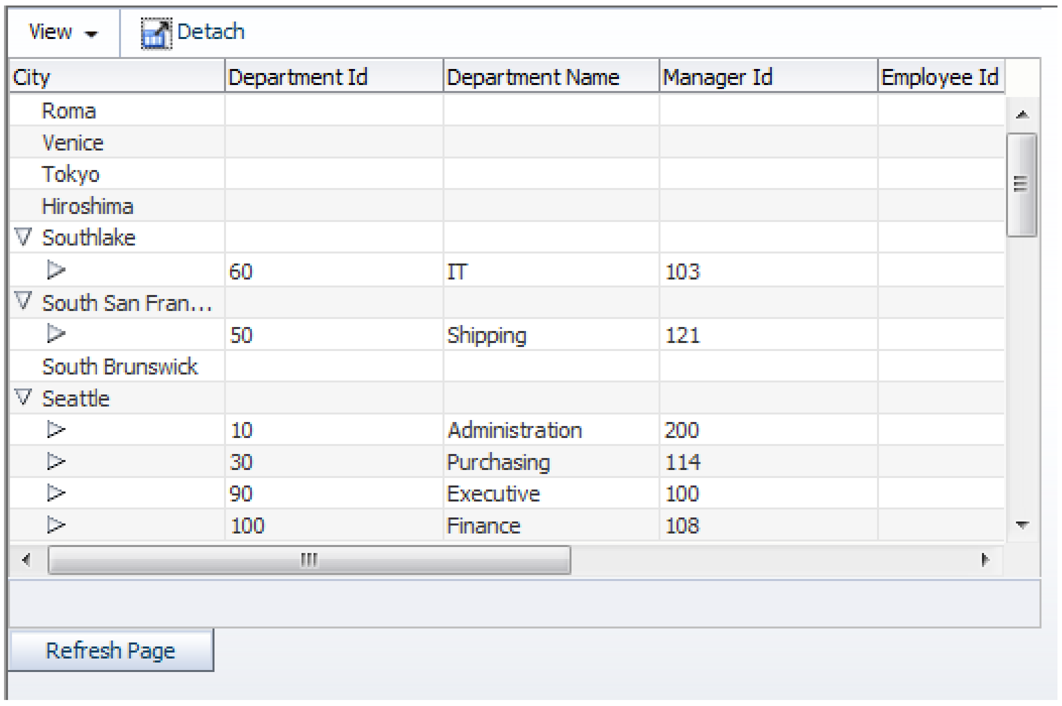
Introduction

The ADF Faces tree component has a disclosed rowKeySet property that can be used to programmatically set the node keys that should display in disclosed state when the tree is refreshed or the view initially renders. In the Oracle JDeveloper 11g R1 PS2 sample for this article, the initial state is pogrammatically defined so that all first level nodes are disclosed.



```
1000 Roma
1100 Venice
1200 Tokyo
1300 Hiroshima
▽ 1400 Southlake
  ▷ 60 IT
▽ 1500 South San Francisco
  ▷ 50 Shipping
1600 South Brunswick
▽ 1700 Seattle
  ▷ 10 Administration
  ▷ 30 Purchasing
  ▷ 90 Executive
  ▷ 100 Finance
  ▷ 110 Accounting
  ▷ 120 Treasury
  ▷ 130 Corporate Tax
  ▷ 140 Control And Credit
  ▷ 150 Shareholder Services
  ▷ 160 Benefits
  ▷ 170 Manufacturing
  ▷ 180 Construction
  ▷ 190 Contracting
  ▷ 200 Operations
  ▷ 210 IT Support
  ▷ 220 NOC
  ▷ 230 IT Helpdesk
  ▷ 240 Government Sales
  ▷ 250 Retail Sales
  ▷ 260 Recruiting
  ▷ 270 Payroll
▽ 1800 Toronto
  ▷ 20 Marketing
1900 Whitehorse
```

The same sample used with a treeTable is shown below. The tree table sample also checks if the RowKeySet has entries and if doesn't set the default.



City	Department Id	Department Name	Manager Id	Employee Id
Roma				
Venice				
Tokyo				
Hiroshima				
▽ Southlake				
▷	60	IT	103	
▽ South San Fran...				
▷	50	Shipping	121	
South Brunswick				
▽ Seattle				
▷	10	Administration	200	
▷	30	Purchasing	114	
▷	90	Executive	100	
▷	100	Finance	108	

The tree data in the sample is queried from an ADF Business Components business service through the ADF binding layer. A phase listener on the enclosing view is used to ensure the row keys for the disclosed nodes is set before rendering. The initial page renders as shown in the image above.

Defining the PhaseListener for the f:view component

In the sample application, all tree nodes of the first tree level are expanded. For this, in a phase listener, the ADF Faces tree component is looked up in the view and the JUCtrlHierBinding reference to the ADF binding layer is obtained through the CollectionModel read from the tree value property. With the ADF tree binding handle, the root node of type JUCtrlHierNodeBinding is accessed and used to determine all the level children. In an iteration, the rowKeys of the child nodes are read and added to a java.util.List, which then is added as an entry to the RowKeySetImpl class that represents the disclosed RowKeySet of the tree component.

```
import javax.faces.component.UIComponent;
import javax.faces.component.UIViewRoot;
import javax.faces.context.FacesContext;
import javax.faces.event.PhaseEvent;
import javax.faces.event.PhaseId;
import oracle.adf.view.rich.component.rich.data.RichTree;
import oracle.adf.view.rich.context.AdfFacesContext;
import oracle.jbo.uicli.binding.JUCtrlHierBinding;
import oracle.jbo.uicli.binding.JUCtrlHierNodeBinding;

import org.apache.myfaces.trinidad.model.CollectionModel;
import org.apache.myfaces.trinidad.model.RowKeySet;
import org.apache.myfaces.trinidad.model.RowKeySetImpl;

…
    public void beforeRenderOnViewLoad(PhaseEvent phaseEvent) {
        if (phaseEvent.getPhaseId() == PhaseId.RENDER_RESPONSE){
          FacesContext fctx = FacesContext.getCurrentInstance();
          AdfFacesContext adfFacesContext =
                            AdfFacesContext.getCurrentInstance();

          //make sure this code is executed on the initial page request only
          boolean isInitialRequest = !adfFacesContext.isPartialRequest(fctx);
          if (isInitialRequest) {
                UIViewRoot viewRoot = fctx.getViewRoot();
                UIComponent tree = viewRoot.findComponent("tree1");

                if (tree != null) {
                    CollectionModel model =
                            (CollectionModel)((RichTree)tree).getValue();
                    JUCtrlHierBinding treeBinding =

                              (JUCtrlHierBinding)model.getWrappedData();
                    JUCtrlHierNodeBinding rootNode =
                               treeBinding.getRootNodeBinding();
                    RowKeySet rks = (((RichTree)tree).getDisclosedRowKeys());
                    if (rks == null) {
                        rks = new RowKeySetImpl();
                    }
                    List<JUCtrlHierNodeBinding> firstLevelChildren =
                              rootNode.getChildren();
                    for (JUCtrlHierNodeBinding node : firstLevelChildren) {
                        ArrayList l = new ArrayList();
                        l.add(node.getRowKey());
                        rks.add(l);
                    }
                    ((RichTree)tree).setDisclosedRowKeys(rks);
                }
            }

        }
}
```

The phase listener itself is referenced in a managed bean from the "beforePhase" method property of the f:view component.

```
<f:view beforePhase="#{treeSampleBacking.beforeRenderOnViewLoad}">
 …
</f:view>
```

RELATED DOCOMENTATION

⊠	
⊠	
⊠	