ADF Code Corner

057. How-to declaratively build a master-detail behavior with DVT components

twitter.com/adfcodecorner

Abstract:

A widely unknown feature in ADF bound ADF Faces DVT components is the ability to act as a master in a master-detail relationship. This feature wasn't always in and in the past developers had to listen for the click event to perform this functionality using Java.

This article explains how to declarative build a master-detail behavior using ADF Faces DVT components.

Author: Frank Nimphius, Oracle Corporation
twitter.com/fnimphiu
16-SEP-2010

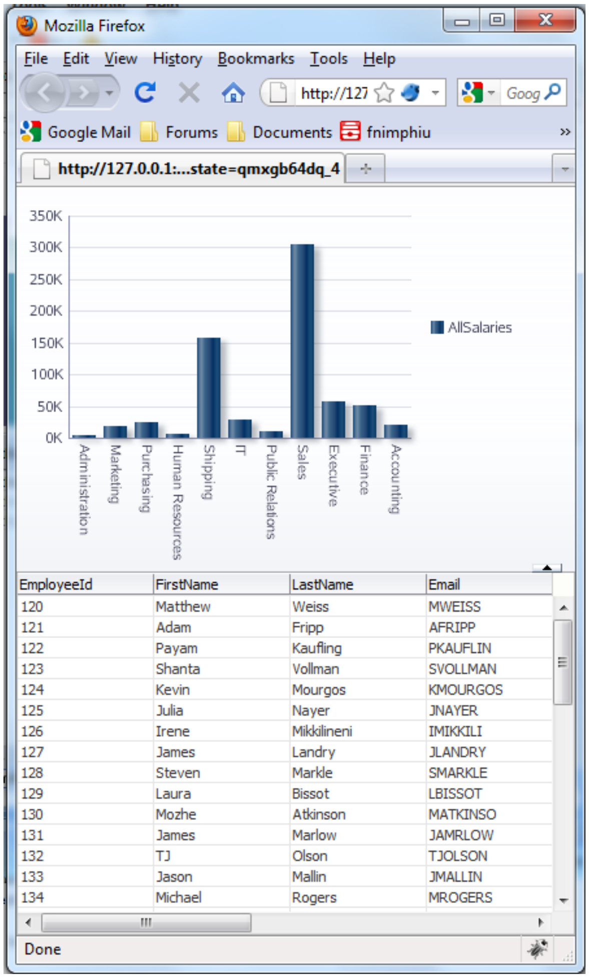
Oracle ADF Code Corner is a loose blog-style series of how-to documents that provide solutions to real world coding problems.

Disclaimer: All samples are provided as is with no guarantee for future upgrades or error correction. No support can be given through Oracle customer support.

Please post questions or report problems related to the samples in this series on the OTN forum for Oracle JDeveloper: http://forums.oracle.com/forums/forum.jspa?forumID=83

Introduction

ADF bound DVT components can be used as a master component in a master-detail relationship. This is not widely known, a good reason to write about it. The outcome of the steps explained in this sample tutorial is shown in the image below



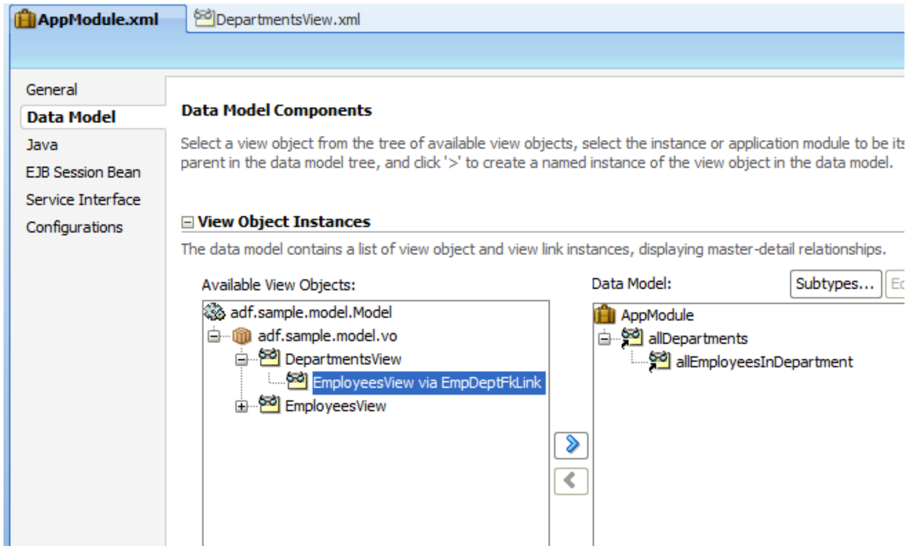
EmployeeId	FirstName	LastName	Email
120	Matthew	Weiss	MWEISS
121	Adam	Fripp	AFRIPP
122	Payam	Kaufling	PKAUFLIN
123	Shanta	Vollman	SVOLLMAN
124	Kevin	Mourgos	KMOURGOS
125	Julia	Nayer	JNAYER
126	Irene	Mikkilineni	IMIKKILI
127	James	Landry	JLANDRY
128	Steven	Markle	SMARKLE
129	Laura	Bissot	LBISSOT
130	Mozhe	Atkinson	MATKINSO
131	James	Marlow	JAMRLOW
132	TJ	Olson	TJOLSON
133	Jason	Mallin	JMALLIN
134	Michael	Rogers	MROGERS

Figure 1: master-detail with DVT component as master

Clicking on a bar in the graph, the table is refreshed showing the details of the employees working in the selected department.

Building the ADF BC model

The ADF BC model is built based on the Departments and Employees table of the Oracle HR schema. You can select the View Object instances in the model view and use the right mouse button to give it a better name, as I did in the sample shown below.



Data Model Components

Select a view object from the tree of available view objects, select the instance or application module to be its parent in the data model tree, and click '>' to create a named instance of the view object in the data model.

View Object Instances

The data model contains a list of view object and view link instances, displaying master-detail relationships.

Available View Objects:

- adf.sample.model.Model
  - adf.sample.model.vo
    - DepartmentsView
      - EmployeesView via EmpDeptFkLink
    - EmployeesView

Data Model:

- AppModule
  - allDepartments
    - allEmployeesInDepartment

Figure 2: ADF BC data model with master-detail relationship

In the Departments View Object, create a transient attribute "allSalary", set its type to "Number" and the value type "Expression". Then add the following Groovy Expression into the value field

```
EmployeesView.sum('Salary')
```

Note that if you used a different database schema, the accessor name "EmployeesView" would be different from the one used in the sample above.

Building the Graph

To build the graph, drag the Department View Object from the DataControl panel to the JSF page

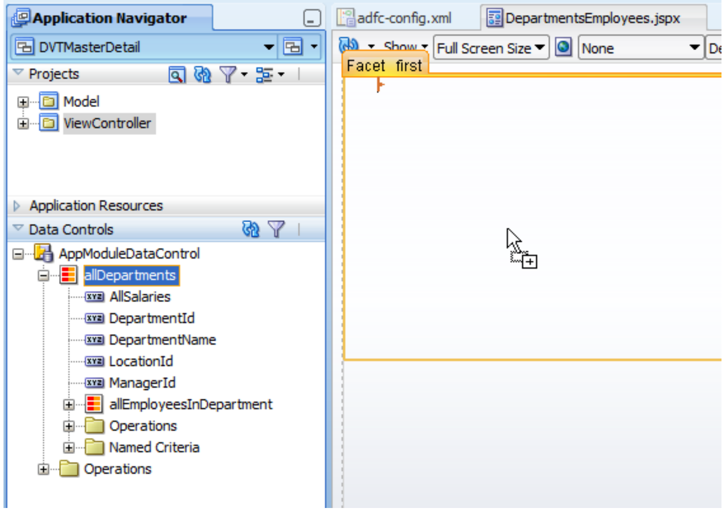


Figure 3: Drag allDepartments as a Graph

From the opened ADF binding menu, choose the Graph option …

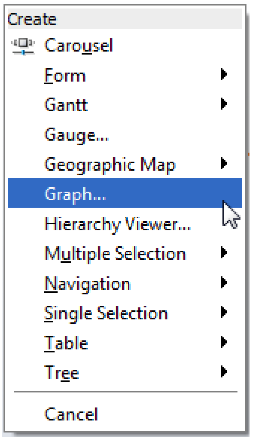


Figure 4: Graph component menu

… and select the graph type of your choice. In this example, the graph type is chosen to be a bar graph.

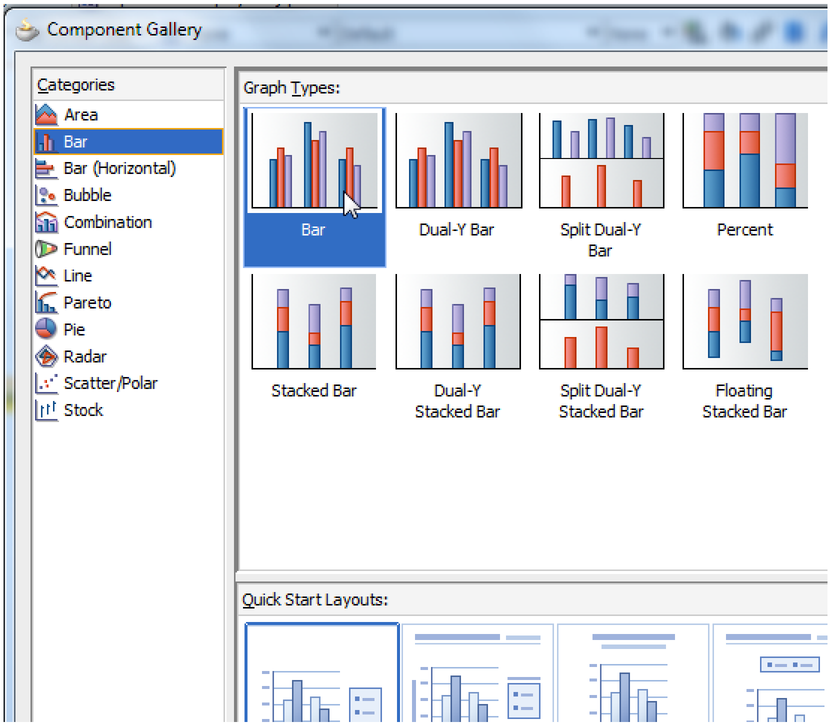
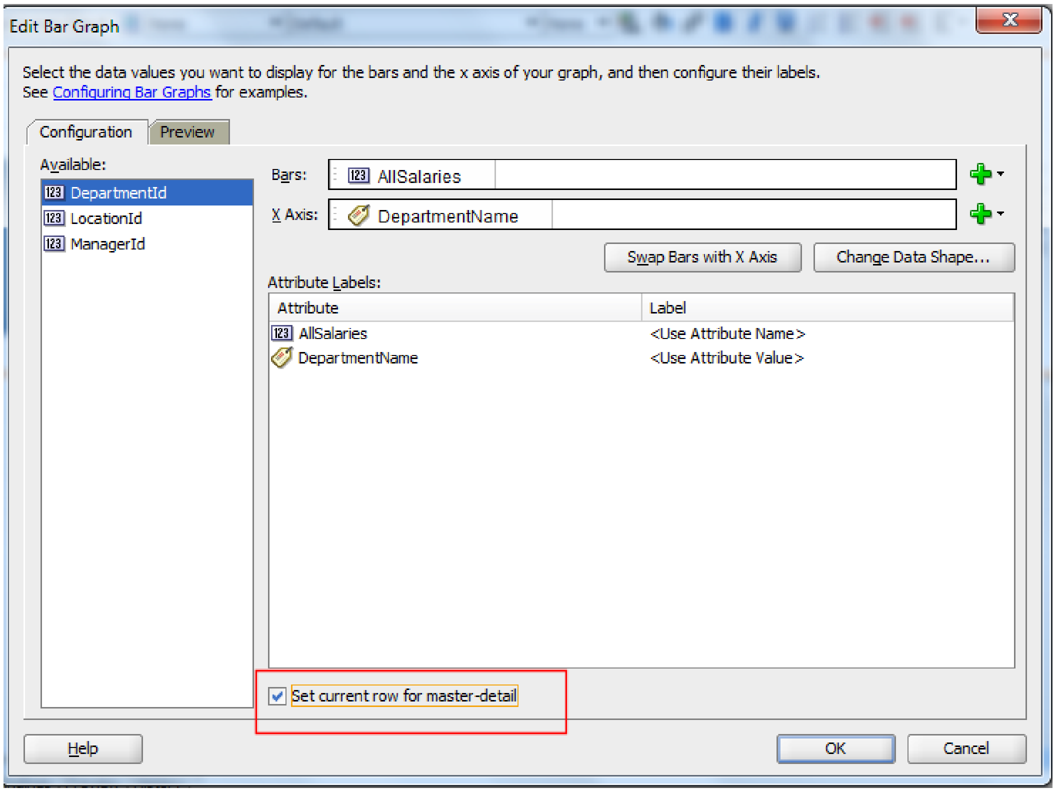


Figure 5: Select a graph type

In the Graph edit dialog, drag the allSalaries transient attribute to the "Bars" field and the "DepartmentName" attribute to the x-Axis.



Select the data values you want to display for the bars and the x axis of your graph, and then configure their labels.
See Configuring Bar Graphs for examples.

Bars: AllSalaries

X Axis: DepartmentName

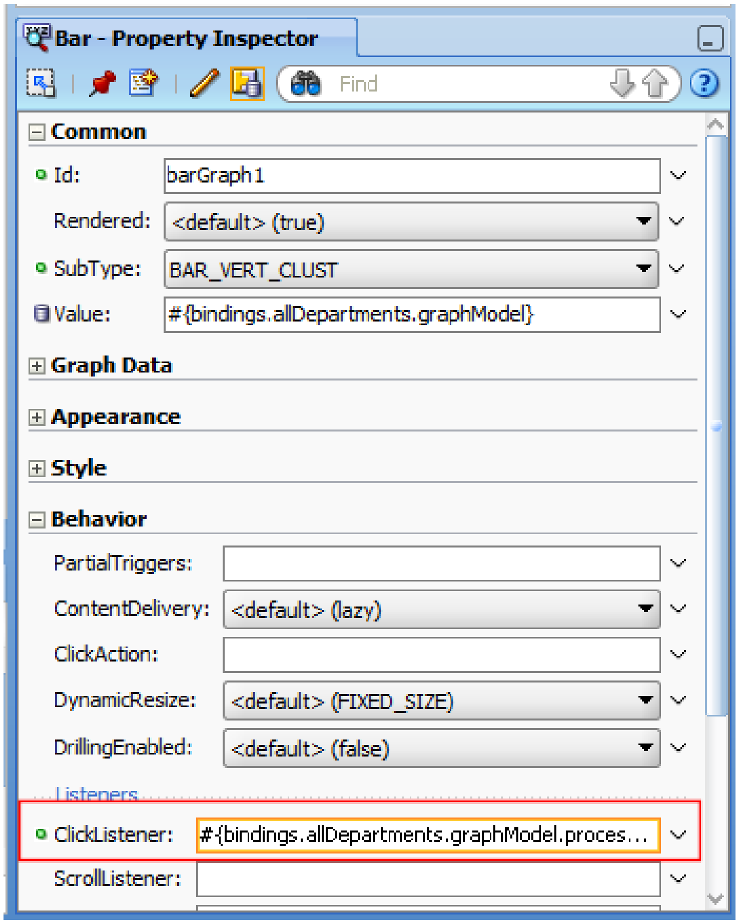
Attribute Labels:

Attribute	Label
AllSalaries	<Use Attribute Name>
DepartmentName	<Use Attribute Value>

☑ Set current row for master-detail

Figure 6: Make sure you select the "Set current row for master ..." option

Make sure you select the "Set current row for master-detail" checkbox, which will configure the graph for the master-detail behavior.



Id: barGraph1
Rendered: <default> (true)
SubType: BAR_VERT_CLUST
Value: #{bindings.allDepartments.graphModel}
PartialTriggers:
ContentDelivery: <default> (lazy)
ClickAction:
DynamicResize: <default> (FIXED_SIZE)
DrillingEnabled: <default> (false)
ClickListener: #{bindings.allDepartments.graphModel.proces…
ScrollListener:

Figure 7: Listener that handles the bar selection

Selecting the "Set current row for master-detail" option adds a click listener configuration to the ClickListener property. The EL string referenced the DVT binding, a FacesCtrl binding class that has a processClick event handler. The processClick handler takes the clicked graph rowKey to set this as the selected row on the binding. This then changes the row currency of the dependent Employees iterator.

To create the dependent Employees iterator, drag the "allEmployeesInDepartment" View Object as a table to the page.

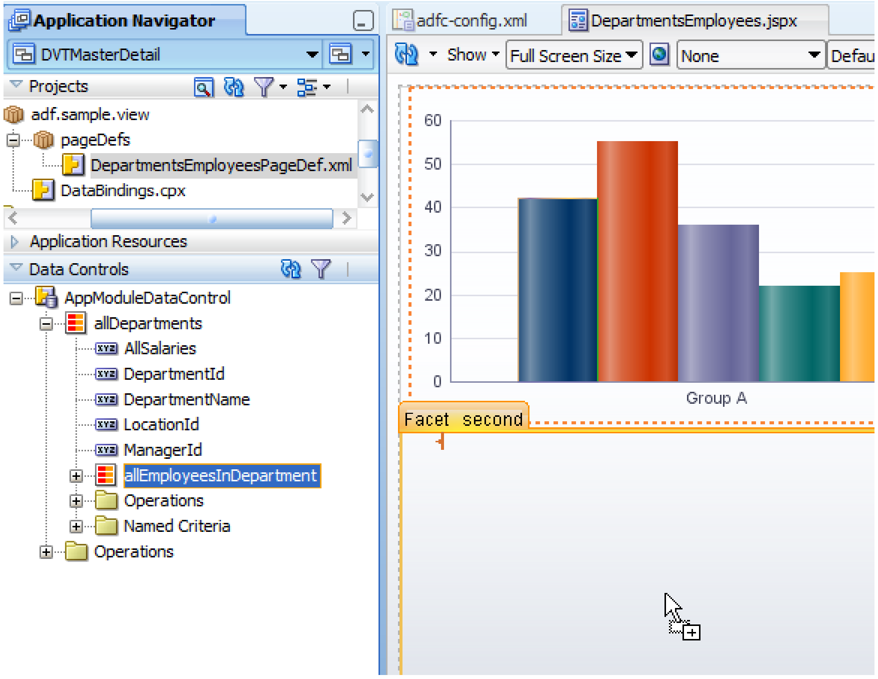
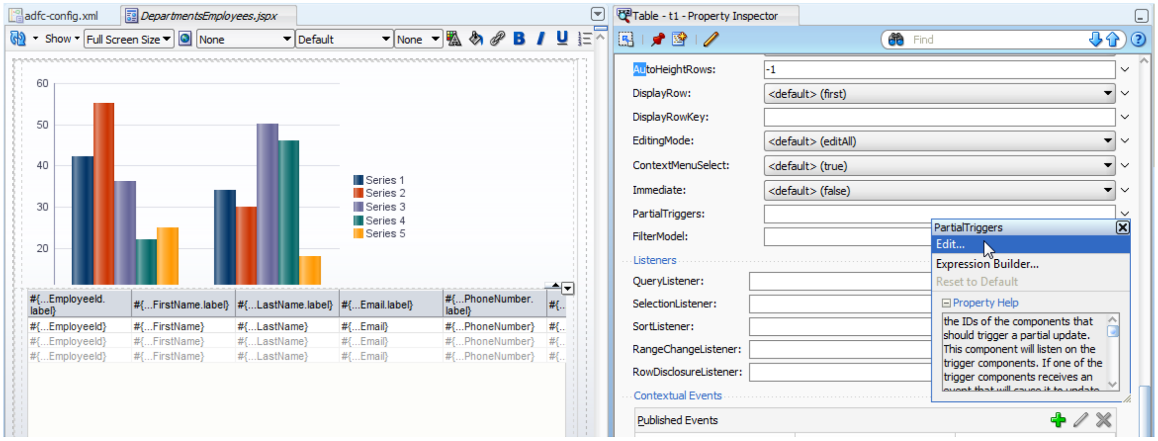


Figure 8: Add a table based on the dependent employee View Object

Select the table component and open the Property Inspector. Select the "PartialTriggers" property and choose the "Edit" option from the context menu.

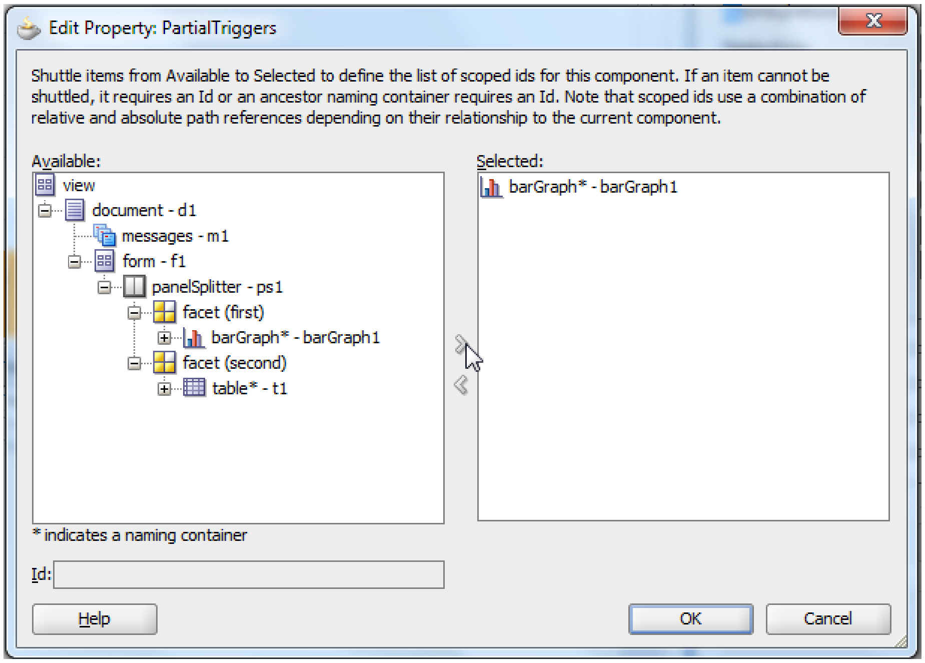


AutoHeightRows: -1
DisplayRow: <default> (first)
DisplayRowKey:
EditingMode: <default> (editAll)
ContextMenuSelect: <default> (true)
Immediate: <default> (false)
PartialTriggers:
FilterModel:
QueryListener:
SelectionListener:
SortListener:
RangeChangeListener:
RowDisclosureListener:

the IDs of the components that should trigger a partial update. This component will listen on the trigger components. If one of the trigger components receives an

Figure 9: Configure the PartialTrigger property for the table

In the opened dialog, select the graph component and press okay after shuttling it to the "Selected" list. The partial trigger setting ensures that a change on the graph always leads to a refresh of the dependent employees table.



Shuttle items from Available to Selected to define the list of scoped ids for this component. If an item cannot be shuttled, it requires an Id or an ancestor naming container requires an Id. Note that scoped ids use a combination of relative and absolute path references depending on their relationship to the current component.

Selected: barGraph* - barGraph1

* indicates a naming container

Id:

Figure 10: Select the graph component to trigger a table refresh

Note: Of course, you can implement your own click listener in a managed bean and programmatically set the current row on the detail iterator. This is what developers did before the "Set current row for master-detail" was added to the dialog

Note: If you forgot to set the "Set current row for master-detail" select box, select the graph component in the visual editor, open the Property Inspector and press the "Edit Component Definition" pencil icon.

Download

The Oracle JDeveloper 11.1.1.3 sample workspace of this article is available on ADF Code Corner: http://www.oracle.com/technetwork/developer-tools/adf/learnmore/index-101235.html

Just make sure you change the database connection of the model to point to a local database of yours that has the HR schema installed

RELATED DOCOMENTATION

☒	
☒	
☒	