ADF Code Corner

Oracle JDeveloper OTN Harvest 05 / 2012

twitter.com/adfcodecorner

Abstract:

The Oracle JDeveloper forum is in the Top 5 of the most active forums on the Oracle Technology Network (OTN). The number of questions and answers published on the forum is steadily increasing with the growing interest in and adoption of the Oracle Application Development Framework (ADF).

The ADF Code Corner "Oracle JDeveloper OTN Harvest" series is a monthly summary of selected topics posted on the OTN Oracle JDeveloper forum. It is an effort to turn knowledge exchange into an interesting read for developers who enjoy harvesting little nuggets of wisdom.

http://blogs.oracle.com/jdevotnharvest/

Author:

Frank Nimphius, Oracle Corporation
twitter.com/fnimphiu
30-MAY-2012

Oracle ADF Code Corner OTN Harvest is a monthly blog series that publishes how-to tips and information around Oracle JDeveloper and Oracle ADF.

Disclaimer: ADF Code Corner OTN Harvest is a blogging effort according to the Oracle blogging policies. It is not an official Oracle publication. All samples and code snippets are provided "as is" with no guarantee for future upgrades or error correction. No support can be given through Oracle customer support.

If you have questions, please post them to the Oracle OTN JDeveloper forum: http://forums.oracle.com/forums/forum.jspa?forumID=83

May 2012 Issue – Table of Contents

JMX ADF Data Control to manage users in WLS LDAP

In his blog, DonatasValys explains how to use the JMX data control in Oracle ADF to administer security in the WLS integrated LDAP server. The blog shows how to create and manage user accounts in the WLS integrated LDAP from with Oracle ADF applications.

http://donatas.nicequestion.com/2012/05/using-jmx-data-control-in-oracle-adf-to.html

Develop with ADF BC for Multiple Databases

A question that comes up once in a while on OTN is how to develop with ADF Business Components for multiple databases. The following paper – written in 2010 – provides guidance for how to develop generic ADF BC applications that you can then run against non-Oracle and Oracle databases.

"How to Develop with ADF Business Components for Multiple Databases" by Blaise Ribet:

http://www.oracle.com/technetwork/developer-tools/jdev/multidatabaseapp-085183.html

25 Great Oracle ADF Resources

John Brunswick from Oracle put together a list of 25 top resources for developers to read that start with Oracle ADF

To quote John's blog:

"For developers new to J2EE the myriad of specifications can be intimidating. This is where Oracle's Advanced Development Framework (ADF) offers enterprise development organizations significant benefit. ADF incorporates the simplicity of visual application design, while anchoring itself in robust J2EE architecture to meet established monitoring and management procedures. Additionally, ADF can be used to expedite WebCenter development, as well as aid in WebCenter template creation.

Here are 25 great resources to help get up to speed on this powerful development technology in the form of Demonstrations, Documentation, Virtual Machines, Integrated Development Environments (IDEs), ADF Tutorials, How-to Blogs, Community Groups and Books."

See what the 25 Resources are reading John Brunswick's blog:

http://www.johnbrunswick.com/2011/05/25-great-oracle-adf-resources/

Warnings that are no errors but Audits

The Oracle JDeveloper IDE alerts developers of missing configurations in the ADF Faces and Java source code by flagging code lines with a red error marker or orange warning marker. For example, the warning

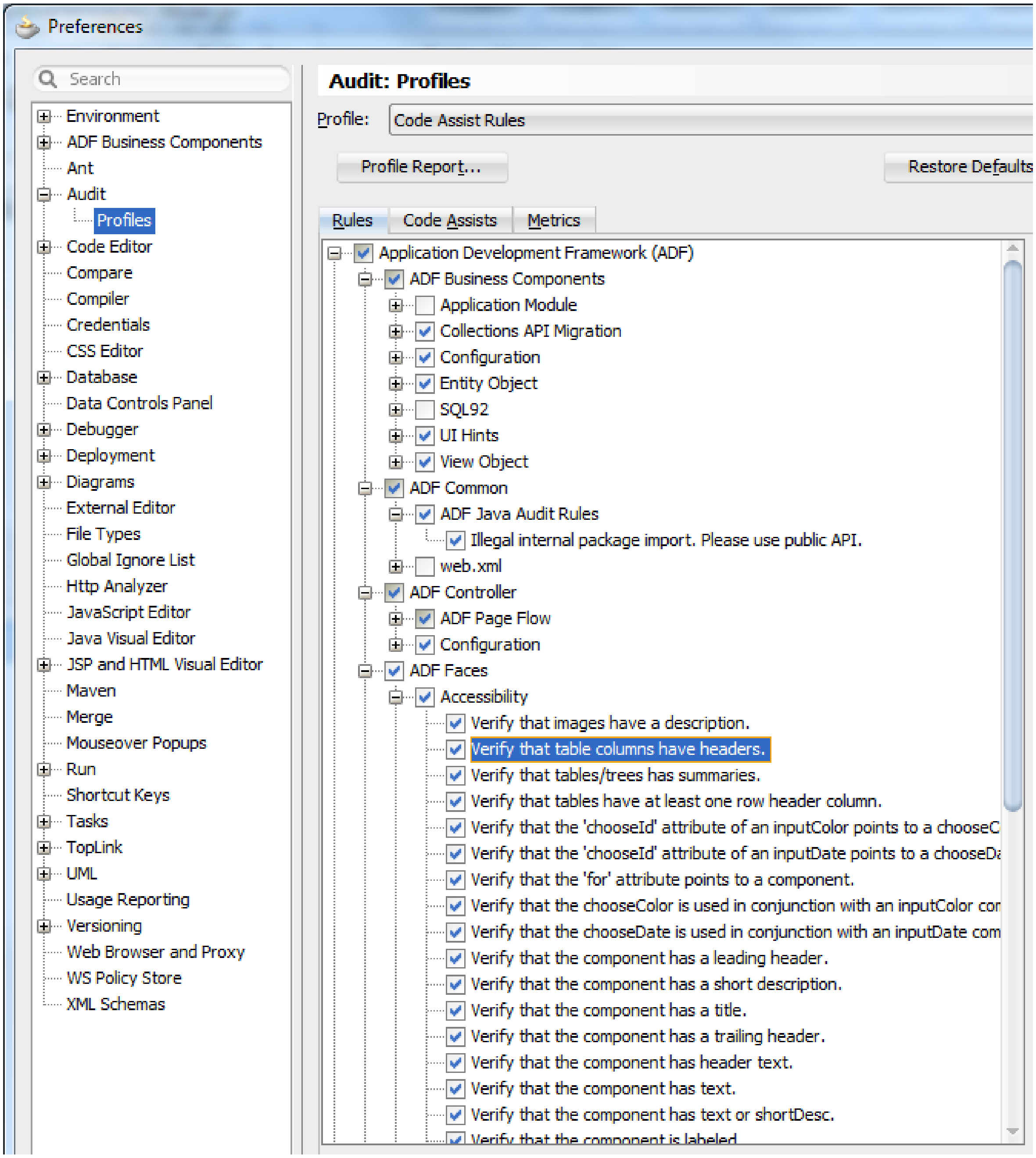
"The table should have at least one column with rowHeader set to true or unstyled, along with child output component"

This message is because of an active accessibility audit rule defined in Oracle JDeveloper to ensure by default ADF application developers implement accessibility features to their applications (Oracle

JDeveloper really is this helpful and friendly). If – for example - accessibility is nothing you want to be bothered with then you can disable the audit rule – like any other rule as well by choosing Tools --> Preferences --> Audit --> Profiles from the Oracle JDeveloper menu. From here you have access to all rules that are defined for web and Java development. For the before mentioned warning, choose

Tools --> Preferences --> Audit --> Profiles --> Accessibility --> "Verify that table columns have headers"

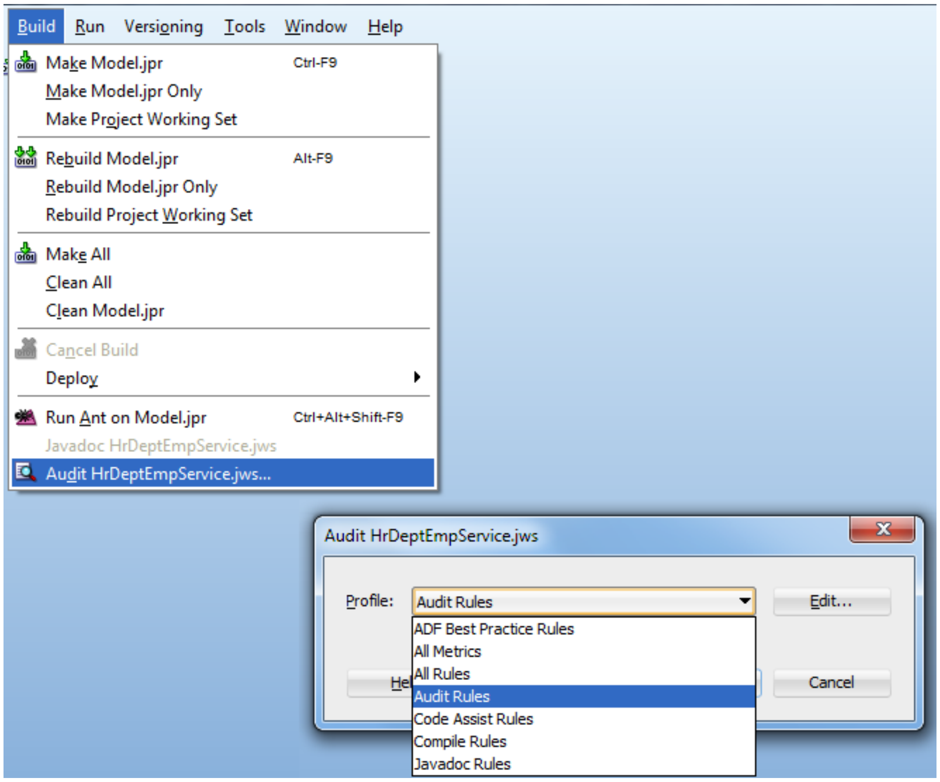
And uncheck the option.



Profile: Code Assist Rules

- ☑ Application Development Framework (ADF)
  - ☑ ADF Business Components
    - □ Application Module
    - ☑ Collections API Migration
    - ☑ Configuration
    - ☑ Entity Object
    - □ SQL92
    - ☑ UI Hints
    - ☑ View Object
  - ☑ ADF Common
    - ☑ ADF Java Audit Rules
      - ☑ Illegal internal package import. Please use public API.
    - □ web.xml
  - ☑ ADF Controller
    - ☑ ADF Page Flow
    - ☑ Configuration
  - ☑ ADF Faces
    - ☑ Accessibility
      - ☑ Verify that images have a description.
      - ☑ Verify that table columns have headers.
      - ☑ Verify that tables/trees has summaries.
      - ☑ Verify that tables have at least one row header column.
      - ☑ Verify that the 'chooseId' attribute of an inputColor points to a chooseC
      - ☑ Verify that the 'chooseId' attribute of an inputDate points to a chooseDa
      - ☑ Verify that the 'for' attribute points to a component.
      - ☑ Verify that the chooseColor is used in conjunction with an inputColor cor
      - ☑ Verify that the chooseDate is used in conjunction with an inputDate com
      - ☑ Verify that the component has a leading header.
      - ☑ Verify that the component has a short description.
      - ☑ Verify that the component has a title.
      - ☑ Verify that the component has a trailing header.
      - ☑ Verify that the component has header text.
      - ☑ Verify that the component has text.
      - ☑ Verify that the component has text or shortDesc.
      - ☑ Verify that the component is labeled

Also worth exploring audit rules that are not enabled by default as there may be some useful rules you like to explicitly run once in a while



Scope object serialization failed (object not serializable)

A common reason for the "SEVERE: ADFc: Scope object serialization failed (object not serializable)" error message is that application developers reference JSF components in managed beans with a scope larger than request. When using the JSF component **Binding** property to point to a managed bean, **think backing bean** and thus don't reference beans that are in scopes larger than request scope (so don't: use viewScope, pageFlowSope, sessionScope or applicationScope).

If, from a managed bean in a scope larger than request you need to have access to a component instance (though the other way around is a more common use case) then either

- look up the component on the UIViewRoot. The Trinidad util class ComponentUtils is a appears to be useful for this too:

  http://myfaces.apache.org/trinidad/trinidad-api/apidocs/org/apache/myfaces/trinidad/util/ComponentUtils.html

- resolve an EL reference to a managed bean in request scope that holds the component binding

As mentioned, a more common use case is to access a bean in a scope larger than request from a backing bean. A use case for this is if you need to remember a specific state, like a list of selected values in a select many list, or the disclosure state of panel tabs.

In this case you would save the state information in a bean with a lifecycle that spans beyond request scope and access it from a request scope backing bean using a managed property or by resolving EL in Java.

Managing component value state correctly

A frequent problem reported on OTN is about failure in saving component values using managed beans especially when multi-select components are used. Usually the problem arises when developers store select choice components values in managed beans they use for JSF component bindings as well (which then makes the managed bean becoming a backing bean). Backing beans however are saved in request scope (at least they should to not err again).

Request scope beans however reset their state after each request, which means that no user update in a select component is persisted beyond the request. The result is that user changes are not displayed in the UI or value change listeners don't execute. The proper way of handling the combination of backing beans and component values saved in a managed bean is to come up with a strategy in which the component value is written to a separate bean in a broader scope (I usually recommend viewScope) than the backing bean. The backing bean (request scope) would then access the bean in the larger scope using a `ValueExpression` or managed bean property to access the user selected data.

I was about to write this solution up in more detail but then found the following blog entry by Duncan Mills:

https://blogs.oracle.com/groundside/entry/the_uimanager_pattern

In his The **UIManager Pattern** blog entry, Duncan documented the same problem from a different angle. The programmer mistake Duncan observed was that component binding references (the Bindings property reference on an ADF Faces component) were stored in managed beans with a scope larger than request, which JavaServer Faces component don't support.

In his write-up Duncan documents the same strategy I was about to document, which is to separate the value saving from the component reference using different managed beans in different scope.

So no need for documenting it twice: https://blogs.oracle.com/groundside/entry/the_uimanager_pattern

Common mistake when iterating <af:table> rows

In a blog entry from 2010, Oracle's Jobinesh Purushothaman writes about exceptions that are getting raised after developers iterate the collection of a RichTable instance using calls to `setRowKey(key)`.

http://jobinesh.blogspot.co.uk/2010/02/common-mistake-while-iterating-through.html

The reason for the exception is that the selected row state is change on the table without synchronization to happen with the ADF binding layer. The failure of synchronizing the table selected row state with the ADF binding layer is because no row selection event is raised on the table if the row currency is changed programmatically. In theory, if you call #{bindings.treebinding.treeModel.makeCurrent}using a method expression in EL

The failed binding synchronization is because no row selection event is fired when the selected row state is changed on the table using Java in a managed bean.

A solution to this problem could be to call #{bindings.treebinding.treeModel.makeCurrent}from a method expression in Java each time you called `setRowKey(key)`. But, assuming the iteration of rows in a table is for aggregating numeric values, or to compare to rows, do you really want to change the current row in the ADF binding layer for each row you need to access?

A better implementation for this use case in my opinion is to read the data from the ADF binding layer instead of the RichTable table instance, which then will not change the row currency at all.

```
RichTable table = ... get table reference ...
Iterator iter = selectedRowKeys.iterator();
CollectionModel model = (CollectionModel ) table.getValue();
JUCtrlHierBinding treeBinding =
                  (JUCtrlHierBinding ) model.getWrappedData();

if (iter != null && iter.hasNext()) {
  Object rowKey = iter.next();
  JUCtrlHierNodeBinding rowData =treeBinding.findNodeByKeyPath(rowKey);
  //Do something here
}
```

Note that the reported problem in Jobinesh's blog is for ADF bound tables configured for single row selection. Still I would prefer my approach over changing the selected row state on a table.

Problem with deleting table rows using ctrl+a for row selection

The following code is commonly shown and documented for how to access the row key of selected table rows in an ADF Faces table configured for multi row selection.

```
public void onRemoveSelectedTableRows(ActionEvent actionEvent) {
   RichTable richTable = … get access to your table instance …
   CollectionModel cm = (CollectionModel)richTable.getValue();
   RowKeySet rowKeySet = (RowKeySet)richTable.getSelectedRowKeys();

   for (Object key : rowKeySet) {
      richTable.setRowKey(key);
      JUCtrlHierNodeBinding rowData =
                            (JUCtrlHierNodeBinding)cm.getRowData();
      // do something with rowData e.g. update, print, copy

    }


    //optional, if you changed data, refresh the table
    AdfFacesContext adfFacesContext =
                    AdfFacesContext.getCurrentInstance();
    adfFacesContext.addPartialTarget(richTable);
    return null;
}
```

The code shown above works for 99.5 % of all use cases that deal with multi row selection enabled ADF Faces tables, except for when users use the ctrl+a key to mark all rows for delete. Just to make sure I am clear: if you use ctrl+a to mark rows to perform any other operation on them – like bulk updating all rows for a specific attribute – then this works with the code shown above. Even for bulk row delete, any other mean of row selection (shift+click and multiple ctrl+click) works like a charm and the rows are deleted.

So apparently it is the use of ctrl+a that causes the problem when deleting multiple rows of an ADF Faces table. To implement code that works for all table selection use cases, including the one to delete all table rows in one go, you use the code shown below.

```
public void onRemoveSelectedTableRows(ActionEvent actionEvent) {
  RichTable richTable = … get access to your table instance …
  CollectionModel cm = (CollectionModel)richTable.getValue();
  RowKeySet rowKeySet = (RowKeySet)richTable.getSelectedRowKeys();
  Object[] rowKeySetArray = rowKeySet.toArray();
   for (Object key : rowKeySetArray) {
     richTable.setRowKey(key);
     JUCtrlHierNodeBinding rowData =
                          (JUCtrlHierNodeBinding)cm.getRowData();
    rowData.getRow().remove();
   }

   AdfFacesContext adfFacesContext =
                   AdfFacesContext.getCurrentInstance();

   adfFacesContext.addPartialTarget(richTable);
}
```

Disabling the initial splash screen

Oracle JDeveloper 11.1.1.6 introduced a new context parameter that suppresses the initial splash screen from being shown. This feature is useful for applications that don't want to show an image while ADF Faces loads the required scripts in the background.

```
<context-param>
 <param-name>oracle.adf.view.rich.SPLASH_SCREEN</param-name>
 <param-value>off</param-value>
</context-param>
```

New support for content delivery networks (CDN)

Oracle JDeveloper 11.1.1.6 supports content delivery networks (CDN) for improved performance when loading resources like application images without developers to change the image reference. The ADF Faces implementation allows one or more prefixes, or "base resource URIs", to be specified for resources. These base resource URIs are defined in the application's `adf-config.xml` file.

The full documentation of this features is available in section A.4.4 Using Content Delivery Networks of the *Fusion Middleware Web User Interface Developer's Guide for Oracle Application Development Framework* product documentation.

http://docs.oracle.com/cd/E23943_01/web.1111/b31973/ap_config.htm#CBDJDAEA

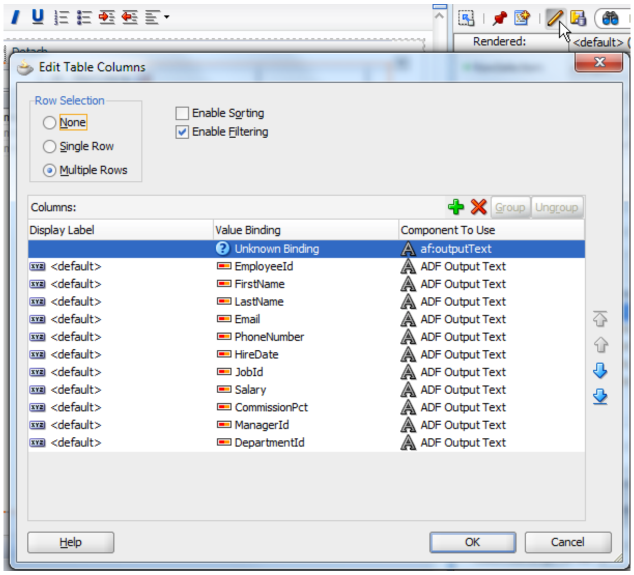
Maiko Rocha from the Oracle Architecture Team (A-Team) did a write up on this topic with an example that shows the performance benefit you gain from this feature

https://blogs.oracle.com/ATEAM_WEBCENTER/entry/adf_faces_11g_ps5_new

Integrated functionality for users to clear table filters

Since Oracle JDeveloper 11.1.1.5, Oracle ADF Faces provides an option for users to clear the table filter content and re-execute the table query. To enable this functionality for an ADF bound table, you

- enable table filters for the table, which you can do either when creating an ADF bound table, or after the fact, using the Property Inspector for the selected table as shown below



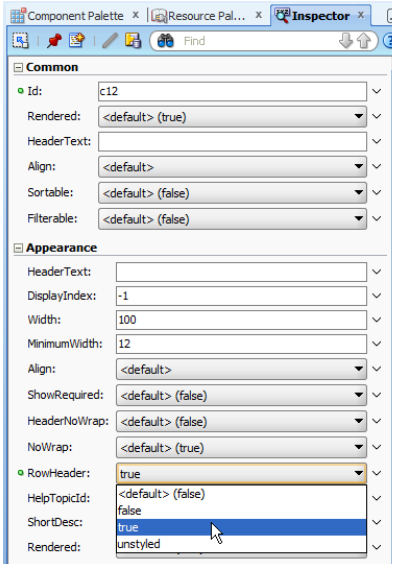
Row Selection
○ None
○ Single Row
● Multiple Rows

□ Enable Sorting
☑ Enable Filtering

Columns:

Display Label	Value Binding	Component To Use
	Unknown Binding	af:outputText
<default>	EmployeeId	ADF Output Text
<default>	FirstName	ADF Output Text
<default>	LastName	ADF Output Text
<default>	Email	ADF Output Text
<default>	PhoneNumber	ADF Output Text
<default>	HireDate	ADF Output Text
<default>	JobId	ADF Output Text
<default>	Salary	ADF Output Text
<default>	CommissionPct	ADF Output Text
<default>	ManagerId	ADF Output Text
<default>	DepartmentId	ADF Output Text

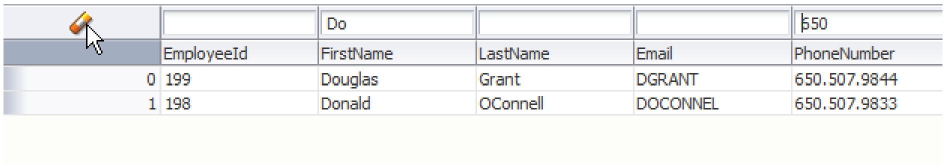
- define a table column to become the row header. The column in the screen shots is transient (in that it is not ADF bound) and shows table row numbers using the #{vs.index} expression in an output text added to the column.



Common
Id: c12
Rendered: <default> (true)
HeaderText:
Align: <default>
Sortable: <default> (false)
Filterable: <default> (false)

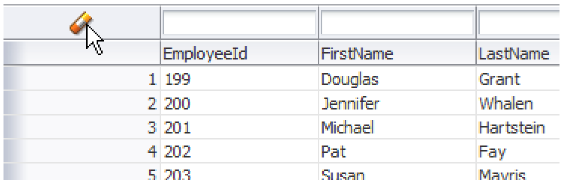
Appearance
HeaderText:
DisplayIndex: -1
Width: 100
MinimumWidth: 12
Align: <default>
ShowRequired: <default> (false)
HeaderNoWrap: <default> (false)
NoWrap: <default> (true)
RowHeader: true
HelpTopicId:
ShortDesc:
Rendered:

The "clear filter" functionality is represented by an icon shown in the table header as shown in the screenshot below. The icon is shown at design- and runtime.



		Do			650
	EmployeeId	FirstName	LastName	Email	PhoneNumber
0	199	Douglas	Grant	DGRANT	650.507.9844
1	198	Donald	OConnell	DOCONNEL	650.507.9833

At runtime, when a user click on the icon, the table filter fields are cleared and the query that is associated with the table is re-executed



	EmployeeId	FirstName	LastName
1	199	Douglas	Grant
2	200	Jennifer	Whalen
3	201	Michael	Hartstein
4	202	Pat	Fay
5	203	Susan	Mavris

To skin the icon, use the skin selector shown below

af|column::filter-clear-all-icon-style

"Styles the icon used for the clearing all filter fields in the table filter row. The icon is specified as a background-image in this key. To use text or an img element instead of a background-image, set -tr-inhibit: background-image in the filter-clear-all-icon-style and define the icon for the filter-clear-all-icon."

Solving File Upload Cancel Issue in JDeveloper 11gR1

In Oracle JDeveloper 11g R1 (I did not test 11g R2) the file upload component is submitted even if users click a cancel button with immediate="true" set. Usually, immediate="true" on a command button by-passes all modle updates, which would make you think that the file upload isn't processed either. However, using a form like shown below, pressing the cancel button has no effect in that the file upload is not suppressed.

```
<af:form id="f1" usesUpload="true">
  <af:inputFile label="Choose file" id="fileup" clientComponent="true"
                value="#{FileUploadBean.file}"
valueChangeListener="#{FileUploadBean.onFileUpload}">
  </af:inputFile>
  <af:commandButton text="Submit" id="cb1" partialSubmit="true"
                    action="#{FileUploadBean.onInputFormSubmit}"/>
  <af:commandButton text="cancel" id="cb2" immediate="true"/>
</af:form>
```

The solution to this problem is a change of the event root, which you can achieve either by setting i) partialSubmit="true" on the command button, or by surrounding the form parts that should not be submitted when the cancel button is pressed with an ii) af:subform tag.

i) partialSubmit solution

```
<af:form id="f1" usesUpload="true">
  <af:inputFile .../>
  <af:commandButton text="Submit" .../>
  <af:commandButton text="cancel" immediate="true" partialSubmit="true"
.../>
</af:form>
```

ii) subform solution

```
<af:form id="f1" usesUpload="true">
  <af:subform id="sf1">
    <af:inputFile ... />
    <af:commandButton text="Submit" ..."/>
  </af:subform>
  <af:commandButton text="cancel" immediate="true" .../>
</af:form>
```

Note that the af:subform surrounds the input form parts that you want to submit when the submit button is pressed. By default, the af:subform only submits its contained content if the submit issued from within.

ADF Code Corner

OTN Harvest Spotlight

- Zeeshan Baig

Zeeshan Baig is a contributor on the Oracle ADF forum on the Oracle Technology Network. Beside of this, Zeeshan is an active blogger, vlogger and Oracle technology enthusiast who loves presenting at Oracle user conferences.

"Since I started my IT career back in year 2000 I have some extreme passion about Oracle's technologies"

- Zeeshan Baig

Blog:	http://www.baigzeeshan.com/
Twitter:	http://twitter.com/baigsorcl

ACC: What is your current role?

ZB: I wear many hats on demand. At the moment I am Team Lead, Architect and Fusion middleware Administrator, managing a WebCenter based environment

ACC: What is your IT background?

ZB: I started my IT career back in 2000 by doing OCP certification in Oracle 8 database administration.

Since then I have worked with different oracle products or tools like database, SQL, PL/SQL, Oracle Forms, APEX and for last few years I am into Fusion Middleware zone working with Weblogic Server, Oracle ADF, WebCenter and SOA.

ACC: How do you currently use Oracle JDeveloper and ADF?

ZB: Currently I use Oracle JDeveloper and ADF to customize WebCenter spaces applications, write portlets and task flows as well.

ACC: So far, what has been your biggest challenge in building Java EE application with Oracle ADF?

ZB: JEE is complex technology. I love ADF because it allows me to build JEE application without knowing much of these complexities. A challenge was that I came from an Oracle Forms development background so I had to spend little time learning Oracle Forms v/s Oracle ADF stuff.

ACC: I guess there are more like you that start from an Oracle Forms background. To help Oracle Forms developers to find a home in Oracle ADF, we provide documentation like this:

http://docs.oracle.com/cd/E23943_01/web.1111/b31974/appendix_formstoadf.htm#CACFEAIJ

However, once you got there, which feature of ADF was the greatest benefit to your project?

ZB: Task flows are so powerful. They are great to build and reuse common code

ACC: Away from the on line help, what have been your most valuable sources of ADF knowledge?

ZB: OTN forums are helpful. Also the ADF community is very active in blogging and I follow blogs, RSS feeds, tweets and the ADF Enterprise Methodology Group (EMG)

ACC: What do you recommend as a starting point and path to learn Oracle ADF?

ZB: Oracle Fusion Developer Guide is best starting point. Grant Ronald's book "Quick Start Guide to Oracle Fusion Development" and Sten Vesterli's "ADF Enterprise Application Development - Made Simple" are valuable resources.

ACC: Are you in any way actively involved in the ADF Community?

ZB: Yes! I regularly visit OTN forums and have a blog at www.baigzeeshan.com. I also run an ADF YouTube tutorial channel at www.youtube.com/baigsorcl

ACC: Recently you presented about Web Center portlets. Can you give us a brief differentiation between portlets and task flows?

ZB: Portlets works with WebCenter. If you don't have WebCenter then ADF task flows are your way to go. Following are few differences between them

1. Portlets could expose and integrate any JSF/ADF based application, part of webpage (web clipping), data source (SQL, CSV, XML) without re-writing code to your application.
2. ADF task flows supports transactions while Portlets do not
3. Communication between task flows done by parameters or contextual events (depends on type) while Portlets uses inter-portlet communication.

ACC: Is there a strict rule for when to use portlets and when not?

ZB: In my opinion if you are integrating external applications outside your domain and want to the reuse the same code then go with Portlets else task flows. An example could be if integrating PeopleSoft with WebCenter where you can expose PeopleSoft as WSRP portlet and use inside your application.

ACC: What is the value add you see in using Web Center compared to coding custom solutions in ADF only?

ZB: The WebCenter is designed for social interaction/collaboration based applications. The core of WebCenter is Oracle ADF. So when using WebCenter in this regard you could save a lot of development effort also you can always extend it as per your requirement.

ACC: What are your top 3 ADF best-practices for ADF developers

ZB:

1) Implement solutions in a declarative way as much as you can. This could save you lot of efforts while debugging and upgrade later.

2) Develop in task flows and use parameters to change the behavior of the same task flow

3) Share ADF Library Jars using artifact repository like Maven, Ivy to avoid confusion with versions

4) Google is your friend.

ACC: What are your top 3 ADF best-practices for Web Center developers in regards to working with ADF?

ZB: 1) Always try to use the framework APIs, never try to hack, it will hit you back later.

2) Use Iterative development feature this could safe lot of time.

3) Use standard HTML and CSS in skins and portlets for browser compatibility.

ACC: ADF Genie grants you a wish, what would you ask for?

ZB: Following are coming into my head at the moment

1) I would love to see something like Oracle Forms APIs like :BLOCK.ITEM or GO_ITEM. At the moment if you want value of an item you have to write 20 lines of code :)
2) Simple built-in PDF generating API could fulfill the common printing reporting requirements.
3) Lightweight free version of ADF framework to build public websites.
4) Free cloud instance for ADF like apex.oracle.com which is available for APEX

ACC: Thank you Zeeshan.