ADF Code Corner

024. How-to build a reusable toolbar with Oracle ADF Declarative Components

twitter.com/adfcodecorner

Abstract:

This article explains how to use Oracle ADF declarative components to build a reusable toolbar that can be used throughout your application to ensure a consistent look and feel. The toolbar has navigation buttons that can be hooked up to the specific data set for which the toolbar is used.

Author: Grant Ronald, Oracle Corporation
17-JUL-2009

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

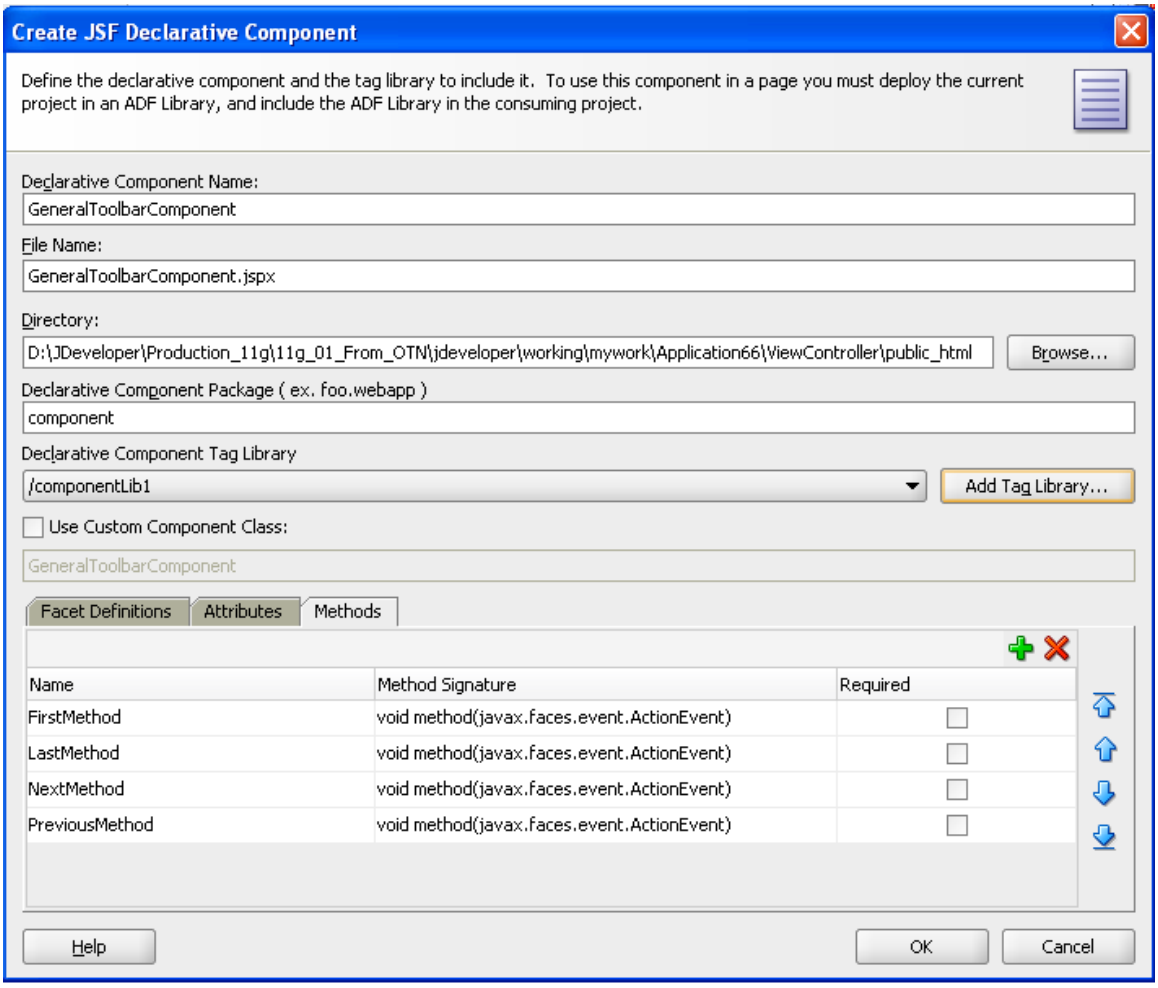
When building an ADF application you will no doubt come across specific cases where a component and functionality could be reused throughout the application. For example, you might decide data pages should have a specific toolbar so that navigation through your data is presented and handled in a specific way. Obviously, each instance of the toolbar would be working on a different data set and so the goal is to create a reusable toolbar that can be parameterised to that it works with different data sets when it it used

Create a declarative component

The first step is to create a declarative component that will implement a reusable toolbar. In a new project **File** -> **New** -> **JSF Declarative Component**. Enter a name for the component (e.g. GeneralToolbar Component) and define a tag library name. This tag library name will be the name of the library in which the component will be stored and is the name you will see in the drop down of the component palette when you come to actually use the component.

You now have to set up the parameters that will be used to customize each instance of the toolbar. In this case you are going to define that you have four "parameters" to this declarative component. Each of these parameters will be a method call to navigate the instance on which the component is used. In the Create JSF Declarative Component dialog, click on the **Methods** tab and enter the names or the parameters: each method relating to a navigation action. You should also define the signature of the method you plan on hooking up to these parameters. For an ADF Action binding (which is what the various Next, Previous actions are) the type is:

```
javax.faces.event.ActionEvent
```

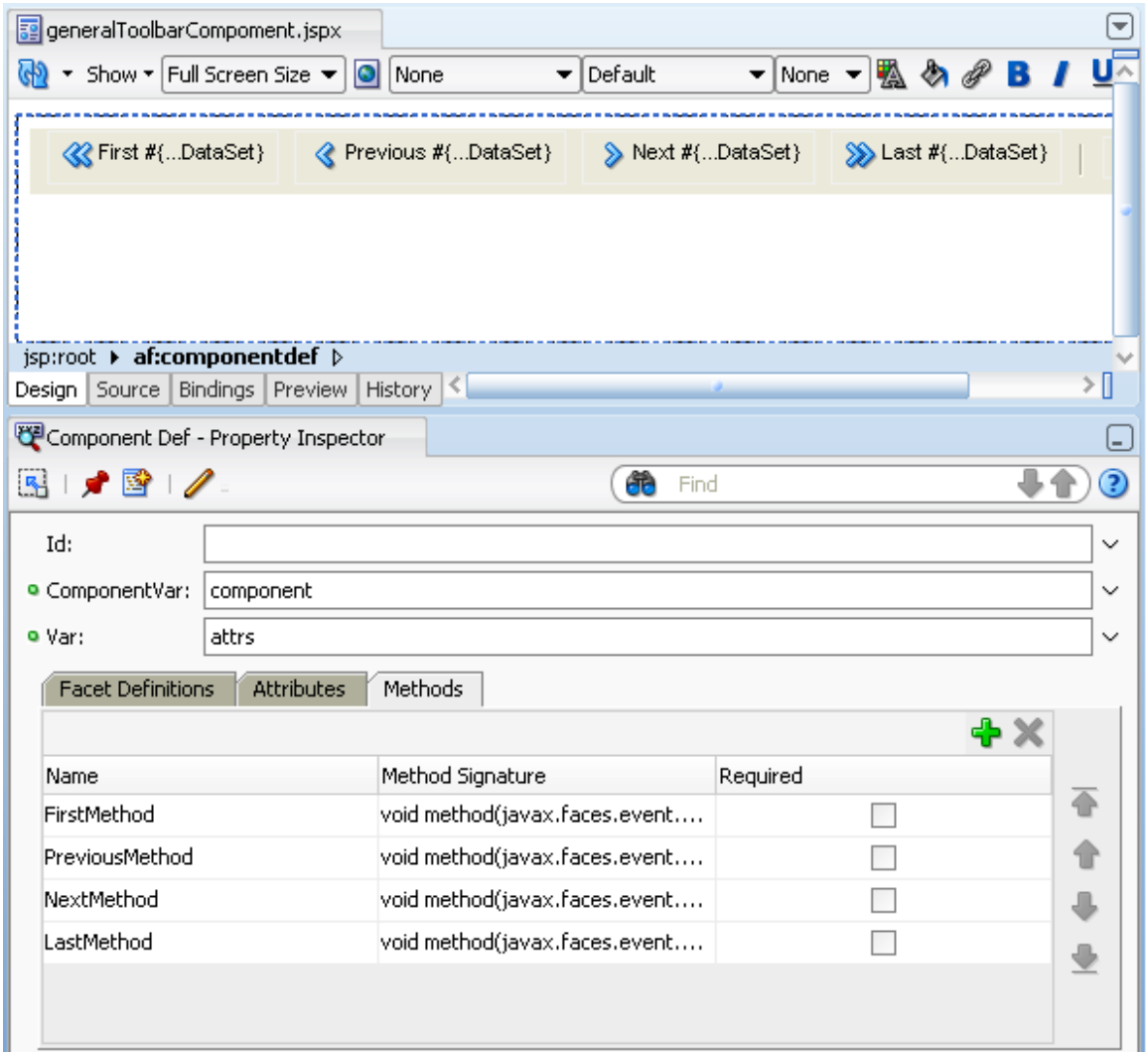


You can optionally define an Attribute (by clicking on the **Attributes** tab) with the name DataSet of type string. This will be used to optionally add a suffix to each button label (so the label could be Next Emp or it could be Next Dept with Emp and Dept being parameters supplied in the toolbar instance)

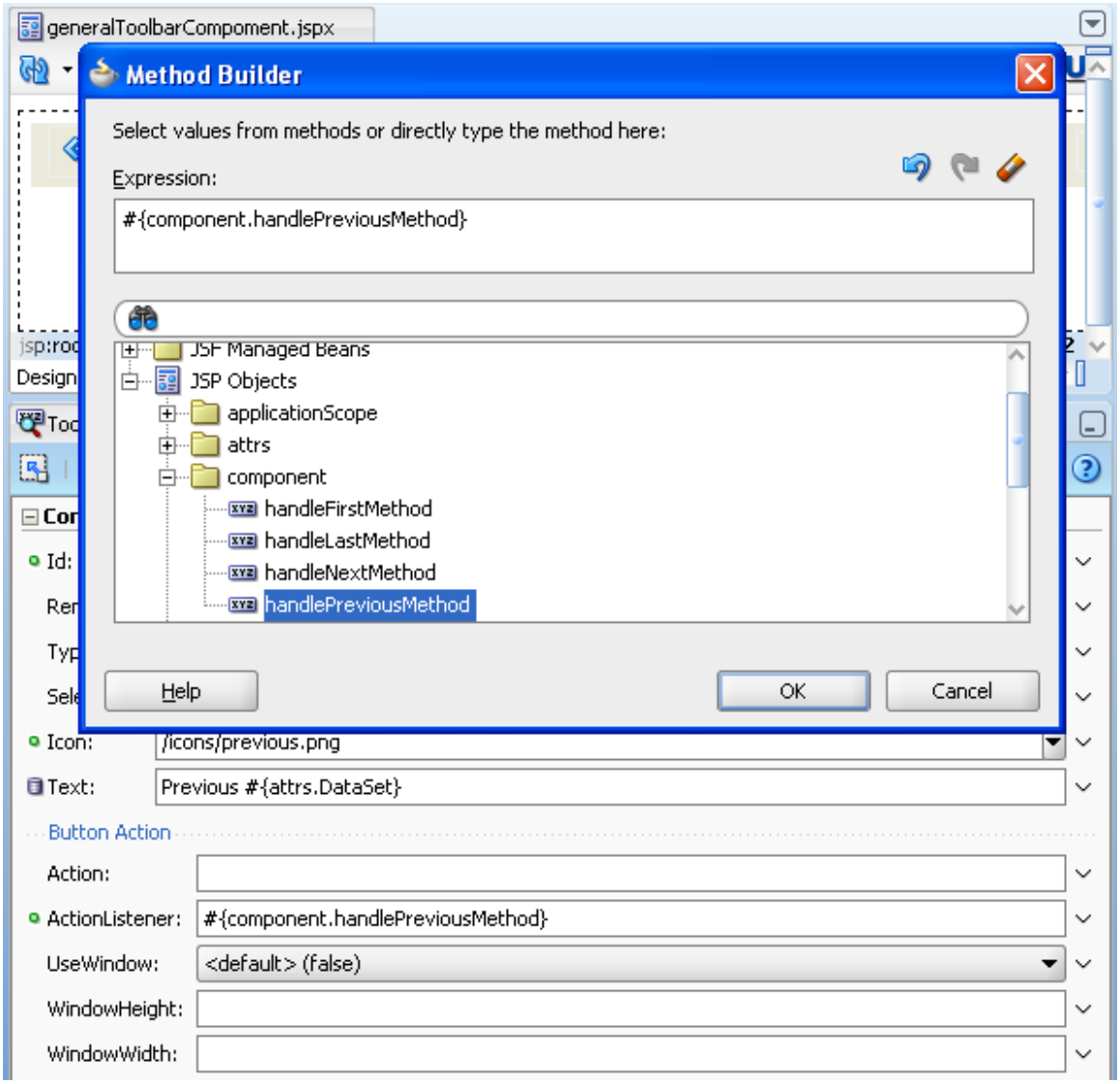
Click **OK** and you can now design the declarative component as you would with any ADF Faces Page.

Design the declarative component

In this case we will keep the toolbar simple, drag on a Toolbar and into the Toolbar drag and drop four Toolbar Buttons. You can optionally add icons to the buttons as well.



The next step is to hook up the method parameters to the specific attributes to which you want them associated. So, in the case of toolbar buttons, this is the ActionListener attribute. Do do this, select the button, and in the property palette go to ActionListener and click on the down arrow to the right and bring up the expression/method builder. Then under **JSP Objects** -> **Components** you will find the method parameters defined earlier. Hook up each button to the appropriate parameter.



You can optionally define the Text property as [a string] JSP Objects -> attrs ->DataSet which is the string parameter you set up earlier.

Deploy the declarative component

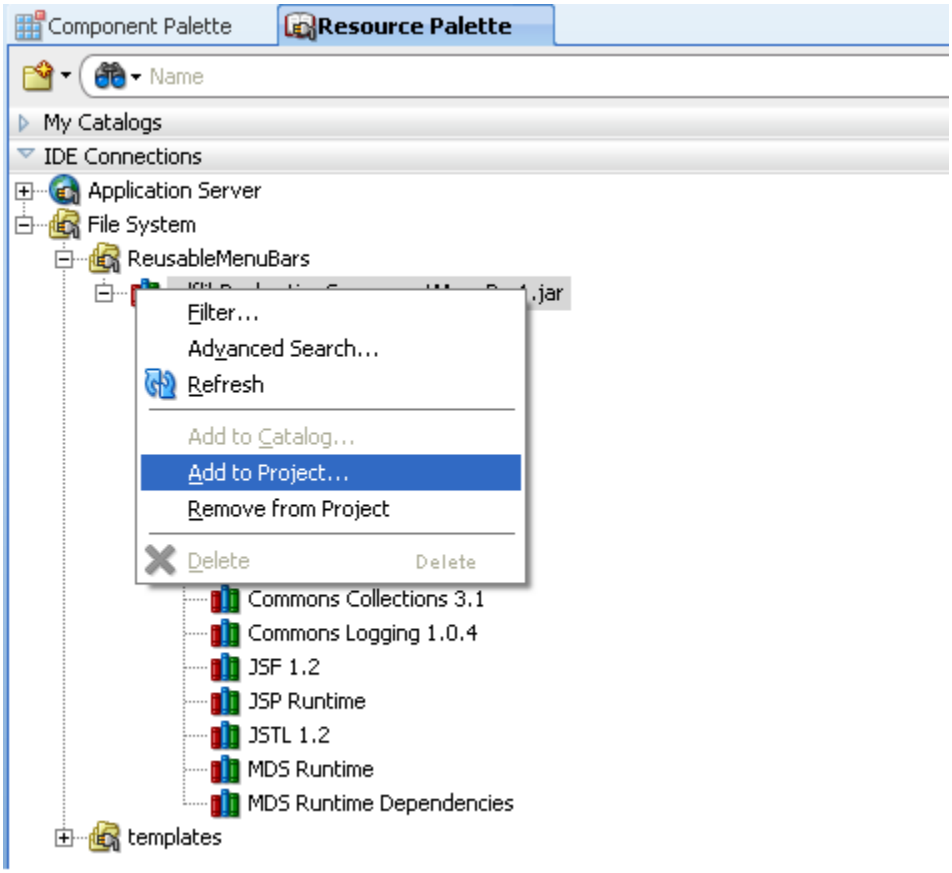
Having built the component you now want to deploy it to a place when it can be picked up and reused, typically by other developers in your team. Right click on the project and select **Project Properties**. Select **Deployment** from the dialog and then press **New**. From the resulting dialog select **ADF Library JAR File** and add a name for the deployment. JDeveloper will now package up the files of your declarative component and prepare it for deployment.

The next step is to actually deploy the JAR file. Select the project, right click and select **Deploy -> [Deployment name] -> to ADF Library JAR**. This will deploy the file to the file system. Note that in the deployment log window you will see the location of where the JAR file is being deployed. Note this.

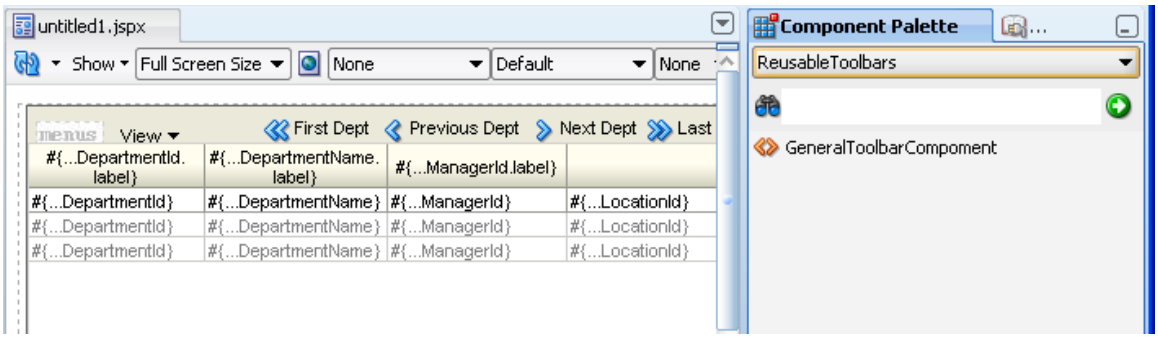
Use the declarative component

Having deployed the reusable component to the file system you (or typically another developer) would want to use that component. From a new application go to the Resource Palette an on the yellow folder click **New Connection** -> **File System**. Enter a name for the connection and in **Directory Path** enter the full path to which you deployed the JAR file (do not include the name of the JAR file itself). Press **OK**. The ADF Library JAR you deployed earlier will now appear in this Resource palette.

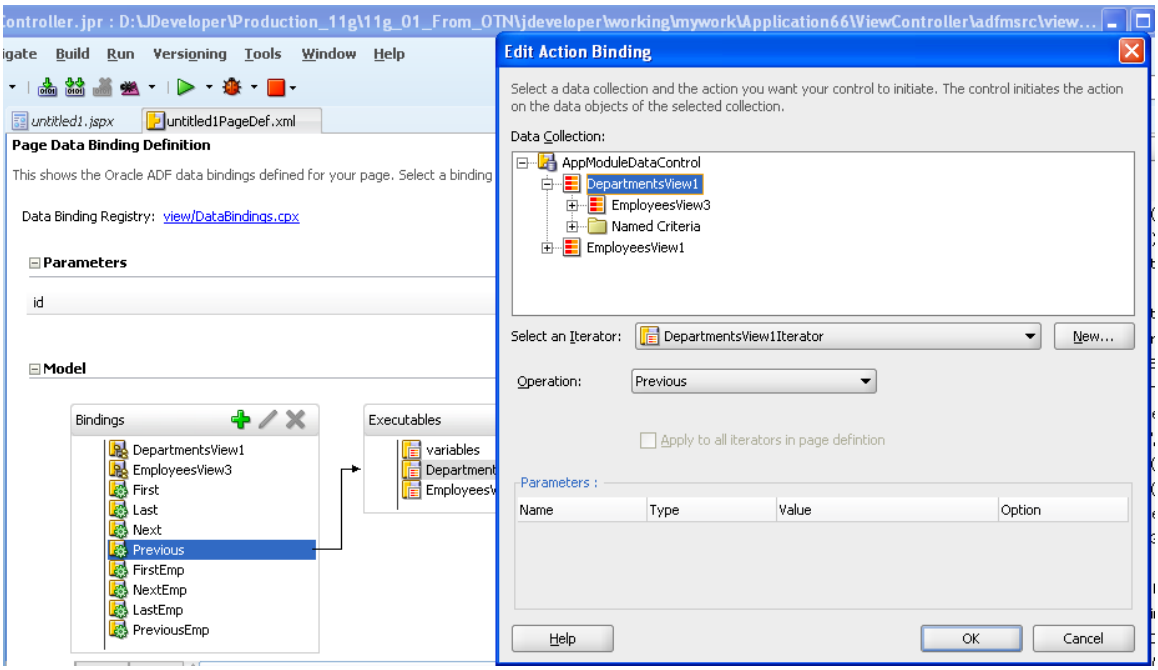
You now want to add that to your new project. Select your project, select the JAR file in the resource palette right click **Add to Project**.



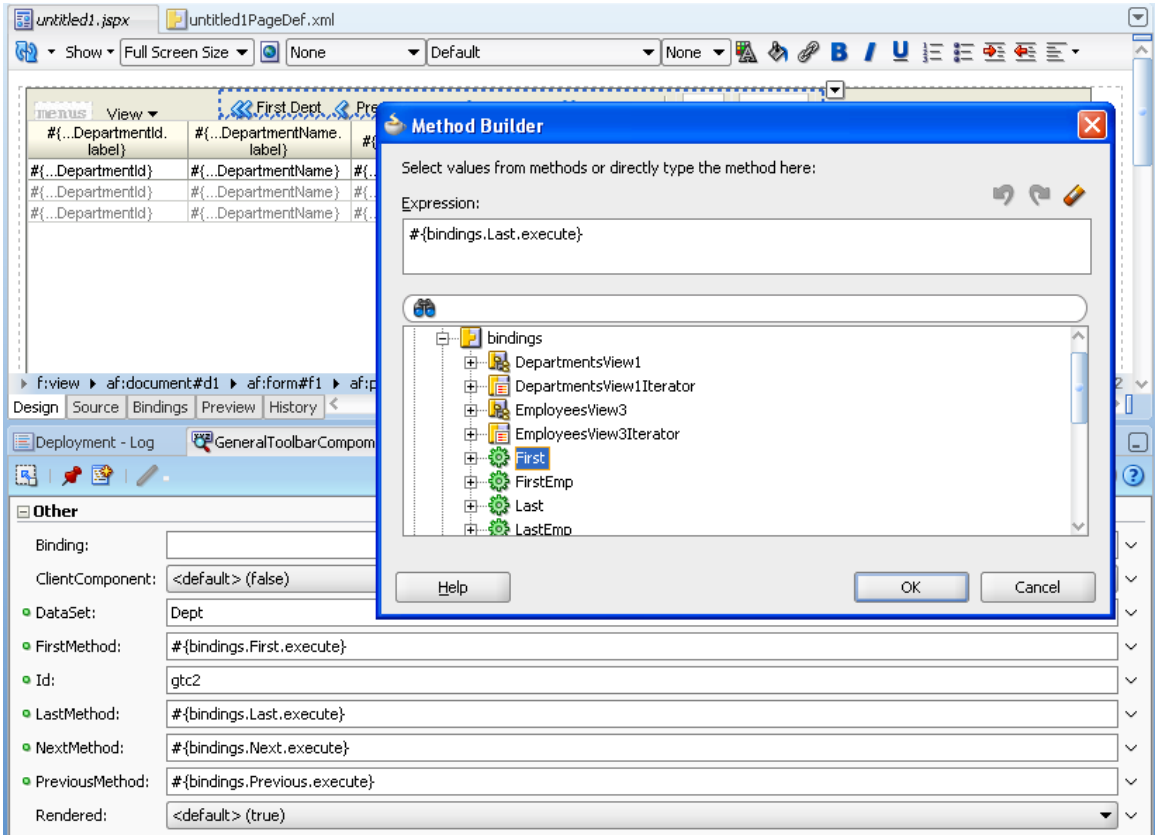
You are now able to start designing your new page. Build your page as you normally would (we'll assume an instance of a Dept table with the generic toolbar) and on the component palette you will now see a new library which includes you toolbar component.



The next step is to hook up the parameters of your toolbar so that they navigate through the Dept table. To do this you must first create the bindings. Right click on the page you are designing and select **Go to Page Definition**. On this page click the green plus sign on **Bindings** and create an **actionBinding**. You then select data collection/iterator and the appropriate operation. You should create action bindings for first, last, next and previous.



The final step is to hook those bindings to the method parameters of the declarative component. Select the toolbar component in the visual editor and in the property inspector you will see the method and attribute parameters you set up when you created the component. Select each of these and using expression language, assign to the appropriate bindings.



With the bindings now assigned to your declarative component you can now run the page and navigate your data set using the generic toolbar.

RELATED DOCOMENTATION

☒	Declarative Components Product Documentation - http://download.oracle.com/docs/cd/E15523_01/web.1111/b31973/af_reuse.htm#CACBFGFC
☒	Oracle Fusion Developer Guide – McGraw Hill Oracle Press, Frank Nimphius, Lynn Munsinger http://www.mhprofessional.com/product.php?cat=112&isbn=0071622543
☒	