
GreenPoint, Inc.

WebCharts3D JavaServer Faces
Component

July 17, 2007

Table of Contents

1.0 INTRODUCTION.....	3
SCOPE	3
2.0 USING JSF TAG.....	4
ADDING WEBCHARTS3D JSF TAG TO WEB PAGES.....	5
PROPERTIES AND DATA BINDING	5
DYNAMICALLY MODIFYING ATTRIBUTES	6
PROCESSING ONCLICK EVENTS.....	6
JSF COMPONENT API.....	7
3.0 USING WEBCHARTS3D WITH SUN JAVA STUDIO CREATOR.....	8
ADDING COMPONENT TO PALETTE	8
ADDING COMPONENT TO PAGE.....	8
EDITING COMPONENT PROPERTIES	9
DEFINING SERVLET PATH	9
4.0 USING WEBCHARTS3D WITH ORACLE JDEVELOPER 10G	10
ADDING COMPONENT TO PALETTE	10
ADDING COMPONENT TO PAGE.....	10
EDITING COMPONENT PROPERTIES	11
DEFINING SERVLET PATH	12

1.0 Introduction

Scope

This document provides WebCharts3D Java Server Faces (JSF) component information and describes WebCharts3D JSF integration with a number of popular IDE.

JavaServer Faces technology simplifies building user interfaces for JavaServer applications. Developers of various skill levels can quickly build web applications by: assembling reusable UI components in a page; connecting these components to an application data source; and wiring client-generated events to server-side event handlers.

WebCharts3D JSF component is a standard JSF component that can be used with any JSF container and integrated into Sun Java Studio Creator, Oracle JDeveloper 10g, Borland JBuilder 2005 Enterprise, Web Sphere Application Studio, and other JSF-enabled IDEs.

2.0 Using JSF Tag

WebCharts3D JSF tag is defined inside wc50.jar in META-INF/wc50jsf.tld. This tag supports the following custom attributes in addition to the standard ones:

action	<p>MethodBinding representing the application action to invoke when this component is activated by the user. The expression must evaluate to a public method that takes no parameters, and returns a String (the logical outcome) which is passed to the NavigationHandler for this application.</p> <p>When this attribute is specified it overrides actions defined in the chart's styles.</p>
actionListener	<p>MethodBinding representing an action listener method that will be notified when this component is activated by the user. The expression must evaluate to a public method that takes an ActionEvent parameter, with a return type of void.</p> <p>When this attribute is specified it overrides actions defined in the chart's styles.</p>
width	Width.
height	Height
type	Image type, i.e. PNG, JPG, SWF, SVG, PDF, TIF, etc.
xmlStyle	Chart's style in xml format.
xmlModel	Chart's model in xml format
servletPath	Path to the bytes serving servlet such as getImage.jsp. The default value of /getImage.jsp?image= requires getImage.jsp page to be placed in the web server's root folder.

Adding WebCharts3D JSF tag to web pages

To use WebCharts3D JSF tag you need to:

1. Setup WebCharts3D on the web server by copying wc50.jar to WEB-INF/lib folder of your web application root.
2. Import wc50jsf.tld taglib with URI <http://www.webcharts3d.com/jsp> into your page by introducing the following statement:

```
<%@ taglib uri="http://www.webcharts3d.com/jsf/v50" prefix="wc3d"%>
```

3. Code the tag by using the imported taglib:

```
<wc3d:chart binding="{PageName.chartName}"  
  action="{provide action or remove this line}"  
  actionListener="{provide action listener or remove this line}"  
  type="PNG" width="400" height="319"  
  xmlStyle="encoded xml style or binding"  
  xmlModel="encoded xml model or binding"  
  servletPath="path to getImage.jsp or similar servlet"  
>
```

4. The default servletPath setting requires getImage.jsp file in the server root folder. If you change the default, for example, to **getImage.jsp?image=**, then you'll need to place getImage.jsp into the same folder(s) where your page(s) that uses this tag is. You can also create a servlet using getImage.jsp as a prototype and specify the path to this servlet.

Properties and Data binding

Since all WebCharts3D JSF tag parameters can be bound there is no any special mechanism for data binding. You can use standard MxChartModel API calls to access files, urls and databases and then bind the result to xmlModel property.

For example, you can define function getChartData in your backing bean as:

```
public String getChartData() {  
    return MxChartModel.load(myConn, "SELECT * FROM Prices", false, true).toXML();  
}
```

and then set xmlModel attribute to `'${PageBeanName.chartData}'`

The same mechanism can be used to define bound chart's styles and other attributes.

Dynamically modifying attributes

In addition to getters and setters for all attributes WebCharts3D JSF component provides getters and setters for two properties not declared in the corresponding tag – widgetStyle and widgetModel. These properties are similar to xmlStyle and xmlModel but instead of taking and returning strings they take and return actual style and model objects. In some cases using object model and its API can be much easier than working with XML.

The following example demonstrates using widgetStyle to toggle the chart's dimensionality:

```
import com.gp.api.styles.chart.*;  
  
.....  
  
public void button1_onClick(ActionEvent ae) {  
    MxChartStyle style =(MxChartStyle) chart1.getWidgetStyle();  
    style.is3D = !style.is3D;  
    chart1.setWidgetStyle(style);  
}
```

If you add a command button to your page that already contains a chart named chart1 and specify button1_onClick as the button's ActionListener, then clicking the button will change the chart's presentation.

Processing OnClick events

WebCharts3D JSF tag component is inherited from UICommand and supports standard action and ActionListener properties. To determine what element of the chart was clicked you can use MxChartComponent's getSelectedRow() and getSelectedCol() properties that return clicked row and column indicies.

NOTE. When you specify either action or ActionListener attribute of WebCharts3D tag, all on click user actions defined in the chart's style are ignored.

JSF Component API

Class `com.gp.api.jsf.MxChartComponent` represents WebCharts3D JSF component. The following table contains the list of the available properties. For the detailed information about this class please refer to JavaDoc.

type	Image type - PNG, JPG, PDF, SWF, SVG, TIF, etc.
width	Image width
height	Image height
xmlStyle	Style in xml format
xmlModel	Model in xml format
widgetStyle	Style object
widgetModel	Model object
selectedRow	Selected row
selectedCol	Selected col
encoding	XML encoding
servletPath	URL to bytes retrieval servlet

3.0 Using WebCharts3D with Sun Java Studio Creator

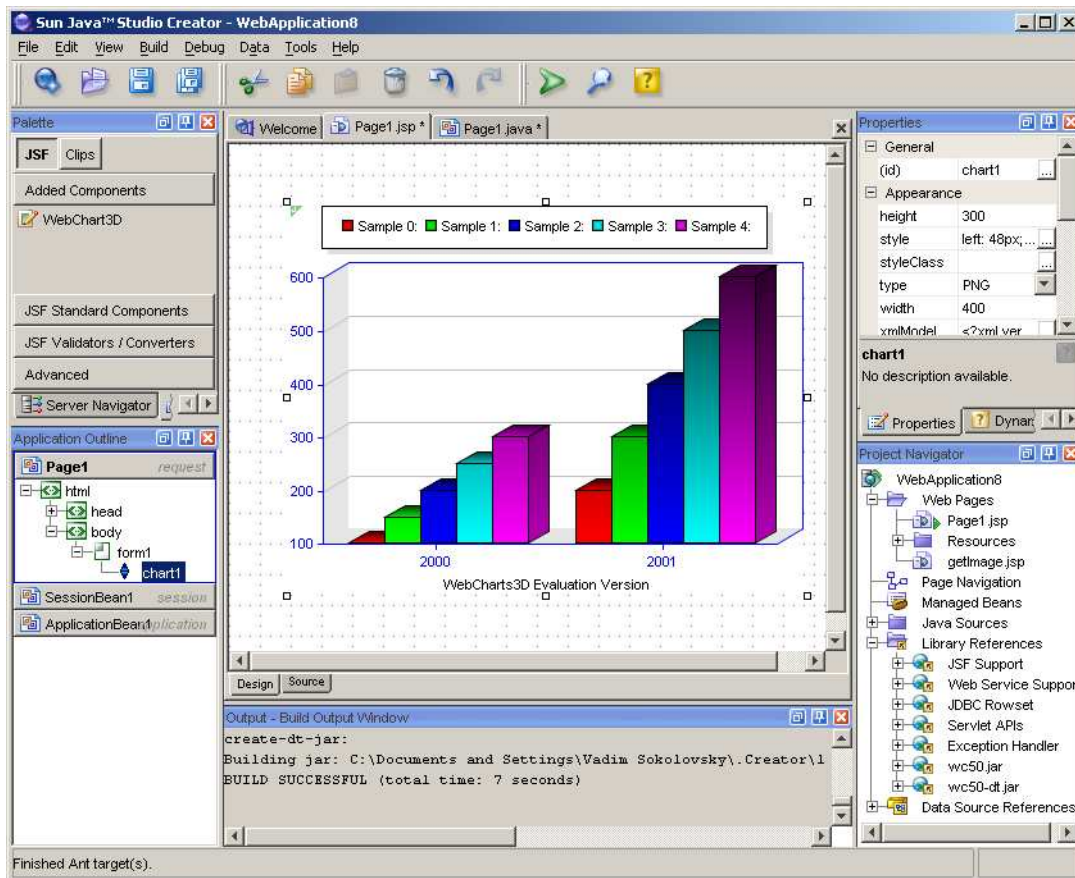
Adding component to palette

To add the component to the palette:

1. In the menu, select Component Library Manager then click Import button located in the bottom left corner of Component Library Manager Dialog.
2. In the Import Components dialog box, enter the location wc50.complib file located in *creator* subdirectory of WebCharts3D installation directory.

Adding component to page

Drag and drop WebCharts3D component into the Visual Editor. The following picture will appear:



Editing component properties

To edit the component properties use Properties pane. The chart's styles and sample data can be editing visually by using visual editors associated with `xmlStyle` and `xmlModel` properties. For the detailed information about xml formats and chart's attributes please refer to the Developer's Guide and built-in WebCharts3D designer documentation. In order open Chart Gallery click on Type combo box icon.

Note. WebCharts3D Design-Time classes do not support `processAction` property in the Events category. To define event handlers you need to enter the correct values into `action` and/or `actionListener` fields into Advanced section of the Properties pane. For example, instead of specifying `OnClick` as `processAction` you need to enter “`{Page1.OnClick}`” into `actionListener` field.

Note. The `action` and `target` attributes in the XML style of the chart will be ignored if `action` and/or `actionListener` properties of the JSF control are defined.

Note. The Design-Time WebCharts3D component always uses PNG as the chart's format and does not show WebCharts3D interactive features. To see these features in action you need to run the project.

Defining servlet path

WebCharts3D JSF tag utilizes indirect method of image delivery and requires path to image retrieval servlet in order to include it into the generated image tag. WebCharts3D provides such servlet in `getImage.jsp` file. You can either use this file or create your own servlet.

The default value of `servletPath` attribute is “`/getImage.jsp?image=`” which means that you need to copy this file to the root of your server. If only a few charts should be displayed, you can specify “`getImage.jsp?image=`” as `servletPath` and then click `File>>Add Existing Item>>JSP source` and add this page to the folder where your pages are located. Note that in this case you'll need to redefine `servletPath` for all charts in your page(s).

NOTE. If you place `getImage.jsp` page into the same folder as your jsf page you might need to redefine `servletPath` parameter to be “`../getImage.jsp?image=`” since all jsf pages are prefixed with `faces/` virtual subdirectory and `getImage.jsp` is a regular JSP page

4.0 Using WebCharts3D with Oracle JDeveloper 10g

NOTE. Current version provides limited support for JDeveloper. We expect to introduce more functionality in the nearest future.

Adding component to palette

To add the component to the palette:

1. In the Tools menu select Manage Libraries. In the opened dialog select JSP Tag Libraries tab.
2. In the tree located on the left side of Manage Libraries dialog select User and then click New button.
3. In the opened file dialog select JSP Tag Library Descriptor, enter full path to wc50.jar located in WebCharts3D installation folder and click Ok.
4. Close Manage Libraries dialog.

For updated information about installing custom components please refer to:

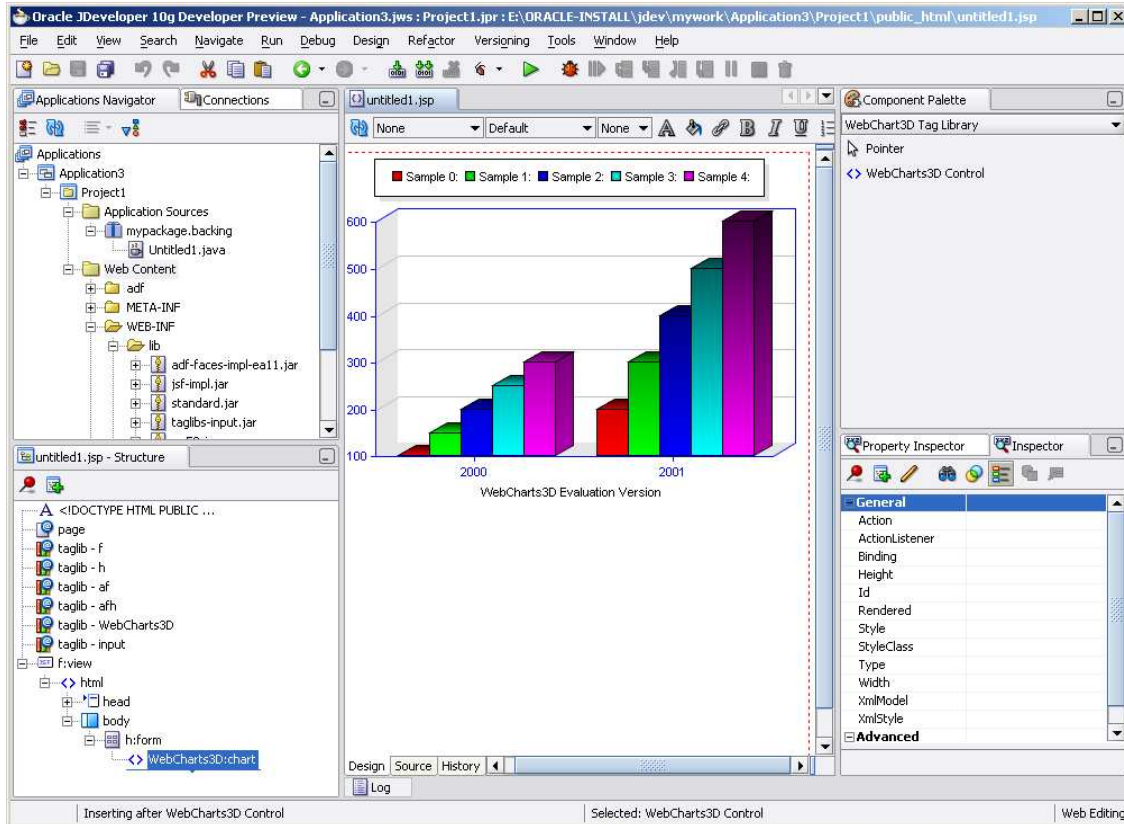
http://www.oracle.com/technology/products/jdev/101/collateral/101/adffaces/howto_customcomponents.html

Adding component to page

1. Create a new JSF page and make sure that WebCharts3D Tag Library is selected as an available tag library.
2. Choose WebCharts3D Tag Library in the Component Palette. Drag and drop WebCharts3D Control into the page.

The following picture will appear:

WebCharts3D JavaServer Faces Component



NOTE. In some cases due to the long initialization cycle the first added chart (or the first viewed chart after JDeveloper is launched) is not displayed in the Visual Editor correctly. Switching to Source view and then back to Design view will refresh the chart.

Editing component properties

To edit the component properties use Properties pane. The chart's styles and sample data can be editing visually by using visual editors associated with `xmlStyle` and `xmlModel` properties. For the detailed information about xml formats and chart's attributes please refer to the Developer's Guide and built-in WebCharts3D designer documentation.

Note. The action and target attributes in the XML style of the chart will be ignored if action and/or `actionListener` properties of the JSF control are defined.

Note. The Design-Time WebCharts3D component always uses PNG as the chart's format and does not show WebCharts3D interactive features. To see these features in action you need to run the project.

Defining servlet path

WebCharts3D JSF tag utilizes indirect method of image delivery and requires path to image retrieval servlet in order to include it into the generated image tag. WebCharts3D provides such servlet in `getImage.jsp` file. You can either use this file or create your own servlet.

The default value of `servletPath` attribute is `"/getImage.jsp?image="` which means that you need to copy this file to the root of your server. If only a few charts should be displayed, you can place the chart into the folder with your jsf pages and overwrite `servletPath` for all charts in your page.

NOTE. If you place `getImage.jsp` page into the same folder as your jsf page you might need to redefine `servletPath` parameter to be `"/getImage.jsp?image="` since all jsf pages are prefixed with `faces/` virtual subdirectory and `getImage.jsp` is a regular JSP page.