Chart Menu

- Introduction
- Chart Menu Properties
  - Configure Chart Menu
  - Configure Datasource
  - Data Binder & Chart Data Mapping
  - Chart Options Properties
  - Advanced Properties
  - Performance & Offline Properties
- Additional Notes & Modifications
  - Display Labels Outside the Pie Chart
  - Hide Gridlines from Chart Plot
  - Interactive Chart
- Download Demo Apps

Introduction

Chart menu allows you to select a form binder or define your own SQL query to display the chart data for the most common graph types. You can also include charts in your userview Dashboard Menu.

Chart Menu Properties

Configure Chart Menu

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Menu element unique id. Userview will use this id in the URL for the menu if the Custom ID is empty.</td>
</tr>
<tr>
<td>Custom ID</td>
<td>Item link slug. Optional field. Value defined here must be unique to the rest of the Userview Menus as the first matching name will be called upon.</td>
</tr>
<tr>
<td>Label</td>
<td>Menu label. Mandatory field.</td>
</tr>
<tr>
<td>Chart Type</td>
<td></td>
</tr>
</tbody>
</table>
  - Area Chart
  - Bar Chart
  - Bubble Chart
  - Candlestick Chart
  - Donut Chart
  - Line Chart
  - Open High Low Close Chart (OHLC Chart)
  - Pie Chart
  - XY Chart |
| Chart Title| Chart Title to be displayed as part of the generated graph. |

Figure 1: Edit Chart Properties
## Configure Datasource

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Screens (Click to view)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Datasource</strong></td>
<td>- Using Data Binder</td>
<td><img src="image" alt="Datasource Configuration" /></td>
</tr>
<tr>
<td></td>
<td>- Default Datasource</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Custom Datasource</td>
<td></td>
</tr>
<tr>
<td><strong>Data Binder</strong></td>
<td>When Datasource is set to &quot;Using Data Binder&quot;, this option will show up.</td>
<td><img src="image" alt="Data Binder Configuration" /></td>
</tr>
<tr>
<td></td>
<td>Advance Form Data Binder has more flexible ways to build chart datasets using join, group, and aggregate functions. Please see <a href="#">Datalist Binder</a> for the available binder to use.</td>
<td></td>
</tr>
</tbody>
</table>
When Datasource is set to use any of the "Datasource", this option will show up. You use an SQL Query to produce the dataset required for the graph type. The first column in the dataset will be assumed for the X-axis/label.

**Example 1:** The first column to be returned from the query must be a label (X-axis), followed by value columns for the Y-axis.

```sql
SELECT c_status AS 'status',
       COUNT(c_status) AS 'count'
FROM app_fd_tix_ticket
WHERE c_status IS NOT NULL
GROUP BY c_status
```

**Example 2:**

```sql
SELECT c.c_claimant,
       SUM( CAST( replace(c.c_total, '$', '') AS DECIMAL(10, 2)) ) AS 'total',
       AVG( CAST( replace(c.c_total, '$', '') AS DECIMAL(10, 2)) ) AS 'avg'
FROM app_fd_hr_expense_claim c
GROUP BY c.c_claimant
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Screens (Click to view)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Order By</strong></td>
<td>Column to be sorted in the graph dataset. This would affect how the graph is plotted.</td>
<td><img src="image1" alt="Figure 3: Data Binder &amp; Chart Data Mapping Properties" /></td>
</tr>
</tbody>
</table>
| **Order**        | • **ASC**  
|                  | • **DESC**           |                         |
| **X-axis Value** | X-axis label.            | ![Figure 3: Data Binder & Chart Data Mapping Properties](image2) |
| **Y-axis Values**| Y-axis dataset.          |                         |

**Chart Options Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Screens (Click to view)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>X-axis Label</strong></td>
<td>X-axis Label</td>
<td><img src="image3" alt="Figure 4: Chart Options" /></td>
</tr>
<tr>
<td><strong>Y-axis Label</strong></td>
<td>Y-axis Label</td>
<td></td>
</tr>
<tr>
<td><strong>Y-axis Prefix</strong></td>
<td>Y-axis Prefix</td>
<td></td>
</tr>
<tr>
<td><strong>Show Legend?</strong></td>
<td>If checked, the legend will be shown in the generated graph.</td>
<td></td>
</tr>
<tr>
<td><strong>Show Value Label in Chart?</strong></td>
<td>If checked, the value label will be shown in the generated graph.</td>
<td></td>
</tr>
<tr>
<td><strong>Stack Series?</strong></td>
<td>If checked, this will affect the generated graph.</td>
<td></td>
</tr>
<tr>
<td><strong>Display as Horizontal Chart?</strong></td>
<td>Display as Horizontal Chart.</td>
<td></td>
</tr>
<tr>
<td><strong>Width</strong></td>
<td>Width in character, for example 100%.</td>
<td></td>
</tr>
<tr>
<td><strong>Height</strong></td>
<td>Height in character, example 300px.</td>
<td></td>
</tr>
</tbody>
</table>
Colors

Series color. Optional field.

Comma-separated values (CSV) of color codes.

**Example 1**: #3333FF,#66FF00,#FF9933,#990000

**Example 2**: red, orange, yellow, green, blue, indigo

---

**Advanced Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>

What are the default colors?

Userview Key Name

When defined, the additional conditions will be appended using the value defined here as the parameter and the userview key value as the value.

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
</table>
| SQL: SELECT category, count(category) FROM table1
| Userview Key Name: type
| Userview Key Value: val
| Resultant SQL: SELECT category, count(category) FROM table1 WHERE type = 'val' |

When userview key value is defined, you may define `{userviewKey}` in your SQL query to have it replaced with the userview key value.

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
</table>
| SQL: SELECT category, count(category) FROM table1 WHERE type = '{userviewKey}'
| Userview Key Value: val
| Resultant SQL: SELECT category, count(category) FROM table1 WHERE type = 'val' |

Custom Header

Custom Header in HTML.

Custom Footer

Custom Footer in HTML.

Performance & Offline Properties

You can configure the Performance settings in this Userview Element which allows one to cache existing content for improved performance and loading speed. Read more at Performance Improvement with Userview Caching.

Additional Notes & Modifications

Display Labels Outside the Pie Chart

The following code can be modified and put in "Custom Header" for displaying the labels outside of the pie chart.
The charts are plotted using jqPlot. Head over to their website here to see the full list of available hooks for customization.

**Hide Gridlines from Chart Plot**

The following code can be modified and put in "Custom Header" property, to hide gridlines from the chart plot.

```javascript
$(function(){
  $.jqplot.preParseOptionsHooks.push(function(args){
    args.seriesDefaults.rendererOptions.dataLabelPositionFactor = 1.05;
  });
});
</script>

**Interactive Chart**

The following code can be modified and put in "Custom Footer" for interactive Charts.

```javascript
$(function(){
  $.jqplot.preParseOptionsHooks.push(function(args){
    args.axesDefaults.drawMajorGridlines = false;
  });
});
</script>
$(document).ready(function() {
    $('#jq_plot_chart').bind('jqplotDataClick', function (event, seriesIndex, pointIndex, data) {
        console.log(event);
        console.log(seriesIndex);
        console.log(pointIndex);
        console.log(data);

        // for chart which used legend and x-axis,
        var xaxis = $('*\:jqplot-xaxis-tick:eq('*pointIndex*')");
        var series = $('*:jqplot-table-legend-label:eq('*seriesIndex*')");

        console.log("x-axis: " + xaxis.text());
        console.log("series: " + series.text());
    });

    // for double click event. Please note the arguments are different.
    $('#jq_plot_chart').bind('jqplotDblClick', function (event, coordinate, points, data) {
        console.log(event);
        console.log(coordinate);
        console.log(points);
        console.log(data);

        if (data) {
            var xaxis = $('*:jqplot-xaxis-tick:eq('*data.pointIndex*')");
            var series = $('*:jqplot-table-legend-label:eq('*data.seriesIndex*')");

            console.log("values: " + data.data);
            console.log("x-axis: " + xaxis.text());
            console.log("series: " + series.text());
        }
    });
});
</script>

This code does not work with OHLC and candlestick charts.

Download Demo Apps

Available in Joget Marketplace

- Candlestick Charting
- Tutorial Apps: Joget Workflow Charts
- Joget DX Showcase ARP_jshowcase_dx.jsa