Mobile Analytics Kit
Code Examples: HTML5

SAP Mobile Platform 2.3

Document ID: DC01729-01-0230-01

Last Revised: January 2013

© 2013 SAP AG or an SAP affiliate company. All rights reserved.
No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries. Please see http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark for additional trademark information and notices.

Contents

Introduction 5

Setting Up Code Examples in SAP Mobile WorkSpace 5

Adding the MAKit HTML5 Library to the Project 7

Code Examples Walkthrough 11

Basic Charts 11

Examples of Chart Configuration 12

ChartTypes_Bubble.xml 12

Drilldown Charts 15

Example of DrillDown Configuration 16

Zooming Charts 18

Example of Zooming Chart Configuration 19

Dynamic Analytic Panel 22

Example of Variable Definition 23

What-If Analysis 25

Example of What-If Analysis Configuration 26

MAKit Charts API 29

Example of MAKit Charts API Methods Script 29

MAKit 2.2 New Feature 31

New Chart API Methods 31

Example of MAKit Charts API Methods Script 31

New Events in MAKit 2.2 34

Example of MAKit Charts Events Script 34

Custom Chart Drawings 35

Example of MAKit Custom Chart Script 36

Line Chart Enhancement 38

MAKit2.3 New Features 38

Overlay Charts 38

Example of Overlay Chart Configuration 39

Example of MAKit Charts API Methods Script 40

Bubble Chart Color 41

Example of Bubble Chart Configuration 42

Example of MAKit Charts API Methods Script 43

Negative Value Support in Pie Charts 45

Example of Negative Value Pie Chart Configuration 45

Charts with Multiple Categories 46

Example of Multiple Categories Configuration 46

Introduction

Mobile Analytics Kit (MAKit) HTML5 is a client-side graphic tool for visualizing and analyzing data through Mobile Web applications on mobile devices (namely tablets) running HTML5-browsers. MAKit HTML5 is delivered as a JavaScript library that can be used with the SAP® Mobile Platform Hybrid Web Container, or any HTML5 Web application.

The MAKit HTML5 code examples in this document demonstrate the core features of MAKit charts: configuring simple charts in normal or full view, configuring drilldown and semantic zooming charts, and applying dynamic analytics and what-if analysis. MAKit HTML5 also supports default color-theme modifications through the use of custom Cascading Style Sheet (CSS) files.

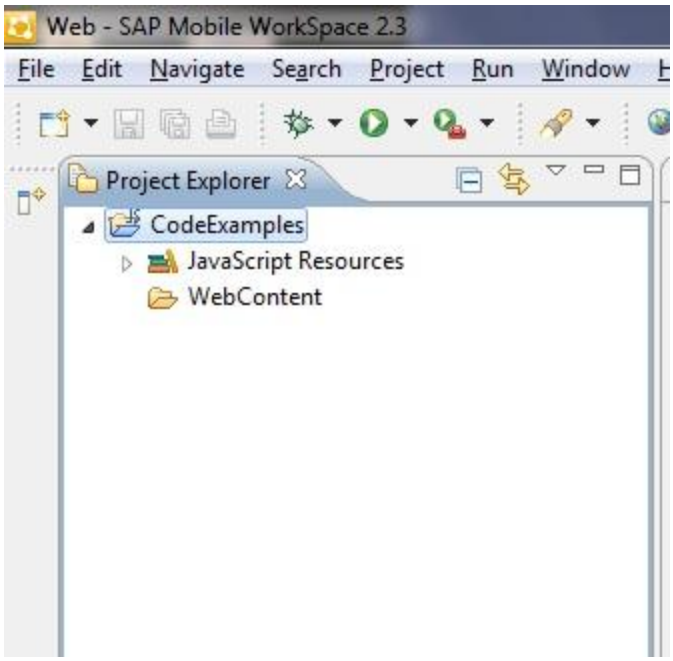
MAKit HTML5 is metadata-driven development tool that can be easily integrated with SAP Mobile Platform or any Web application. An XML-based metadata file defines all the runtime behaviors of MAKit charts. The metadata is passed from the application to the MAKit core engine when MAKit is initialized at runtime. MAKit works on any HTML5-compliant browsers both in desktop and mobile devices.

Setting Up Code Examples in SAP Mobile WorkSpace

Note: These instructions for creating a static Web project are specific to SAP Mobile WorkSpace. If you are using different tools for Web development, the steps may differ.

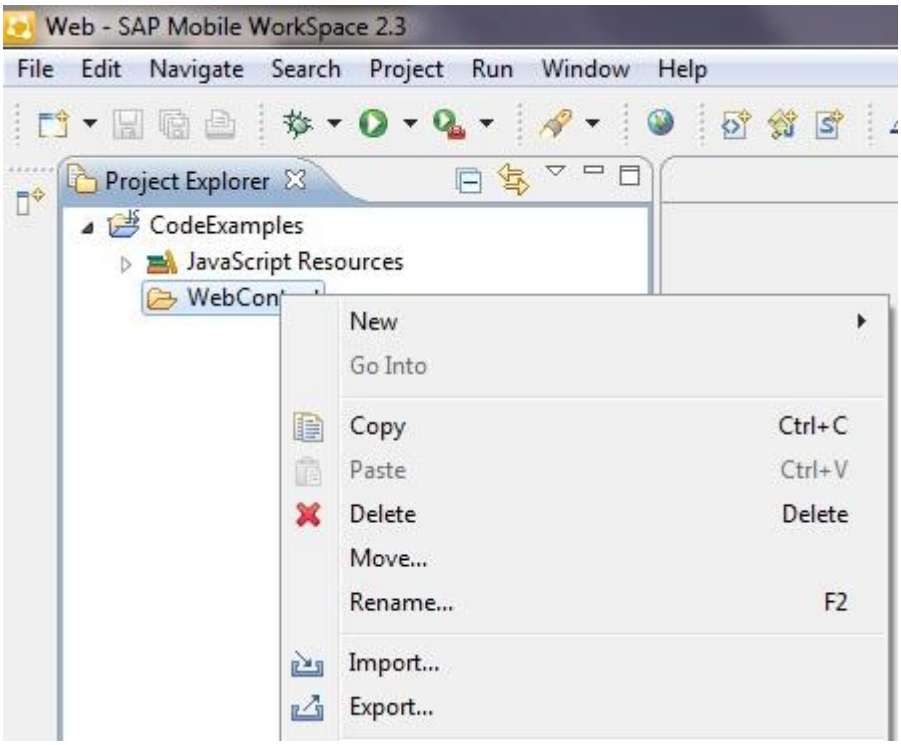
1. Start the SAP Mobile WorkSpace IDE.
2. Select **File** > **New** > **Project.**
3. Select **Web** > **Static Web Project**, then click **Next**.
4. Enter `CodeExamples` as the project name, and unselect "Use default location." Browse to `C:\HTML5` as the location, and click **Next**.
5. Accept the defaults in the next screen by clicking **Finish**.

   The project should look similar to this in Mobile WorkSpace:

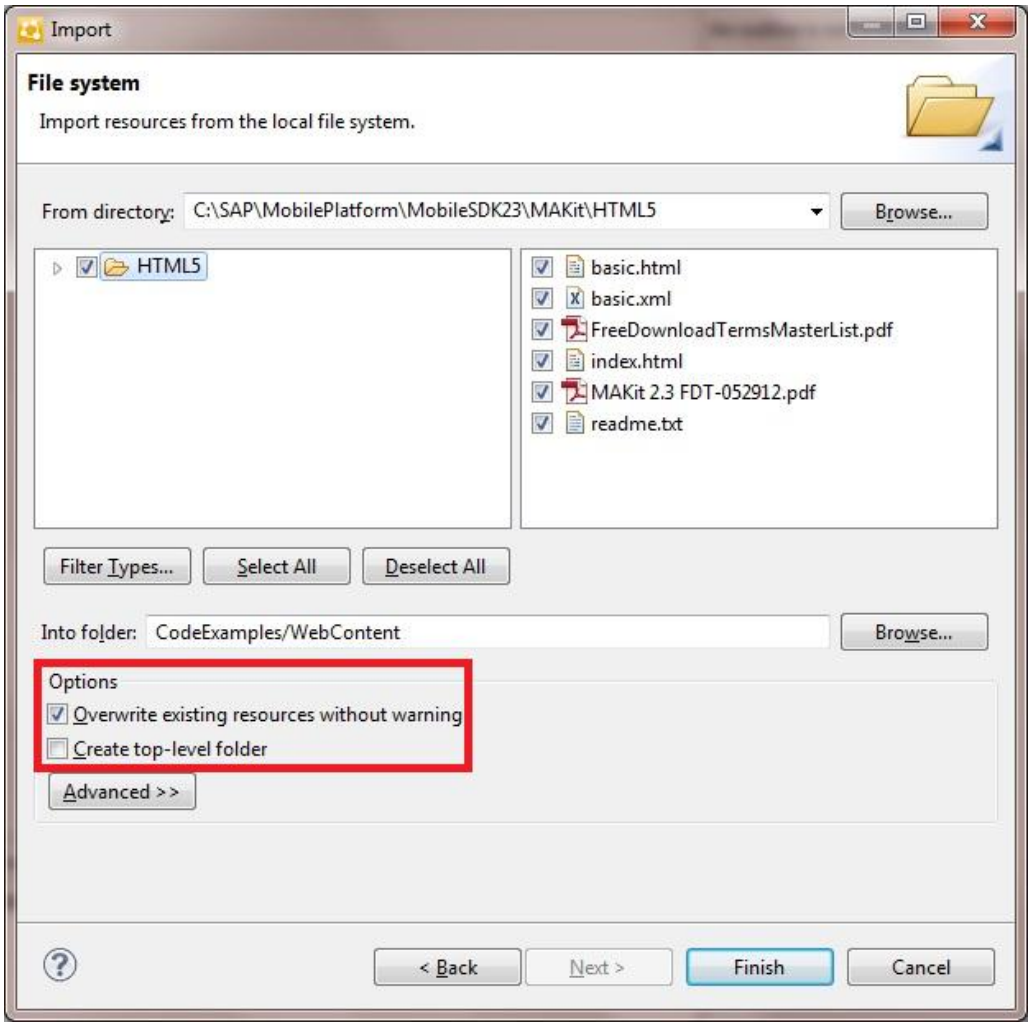

Web - SAP Mobile WorkSpace 2.3
File
Edit
Navigate
Search
Project
Run
Window
Project Explorer
CodeExamples
JavaScript Resources
WebContent

Adding the MAKit HTML5 Library to the Project

1. In SAP Mobile WorkSpace, open the CodeExamples project.
2. In the Project Explorer, expand the `CodeExamples` project folder.
3. Right-click **WebContent**, and select **Import.**

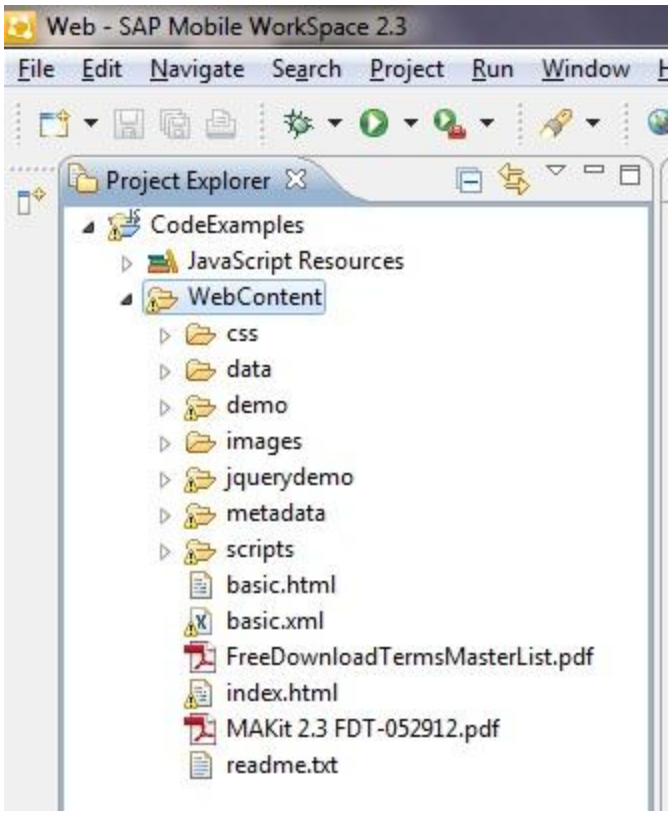


4. In the Import dialog, select **File System** as the import source under the General folder, and click **Next**.



5. In the File System dialog, for the "From directory," browse to `%SAP_MOBILE_INSTALLED%\MobilePlatform\MobileSDK23\MAKit\HTML5`.
6. Click **Select All,** then select **Overwrite existing resources without warning,** unselect "Create top-level folder," then click **Finish**.

   This step imports all the code example contents into the `WebContent` folder. The project should now look similar to:



By default, when you run HTML5 files in Mobile WorkSpace, they are rendered in internal Web browsers within the IDE. To override the default and run the files on Google Chrome or Apple Safari:

a. Verify that the appropriate browser (Google Chrome or Safari) is installed on your desktop.

b. Select **Window** > **Preferences**. In the Preferences dialog, select **General** > **Web Browser**.

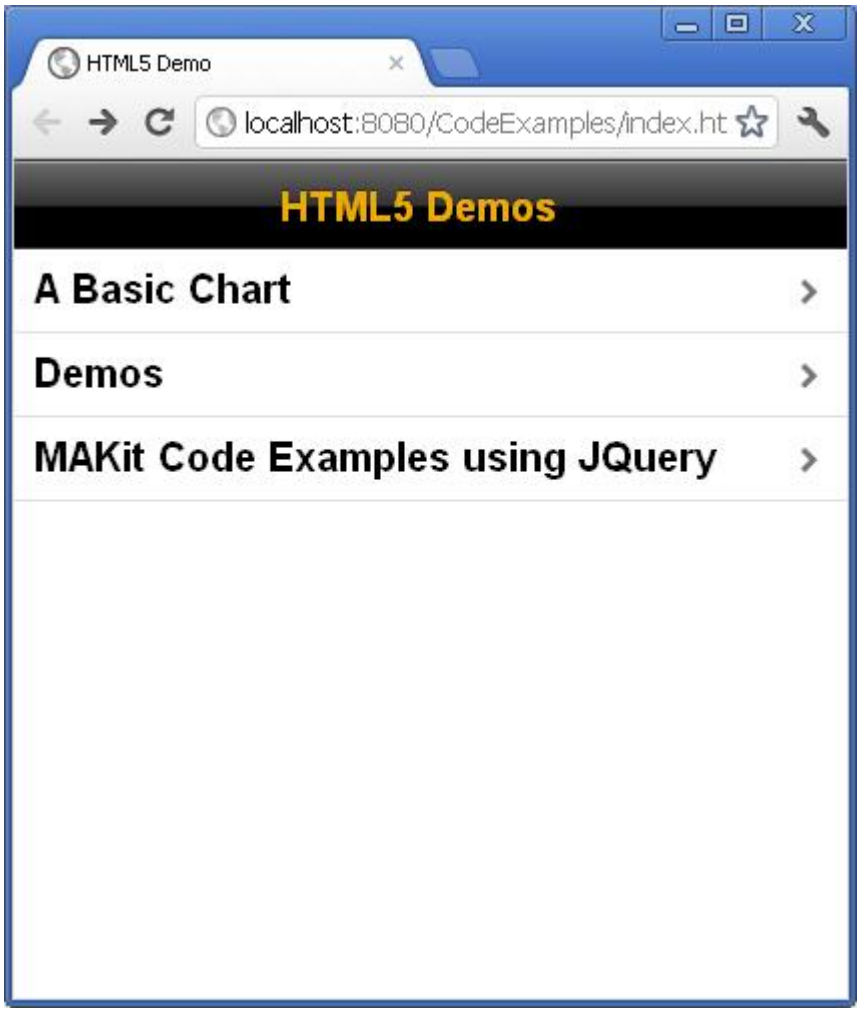
c. Select **Use external browser.** If Google Chrome or Safari does not appear, click **New** and follow the onscreen instructions to add an external browser to the list.

d. Select **Google Chrome** or **Safari** and click **OK**.

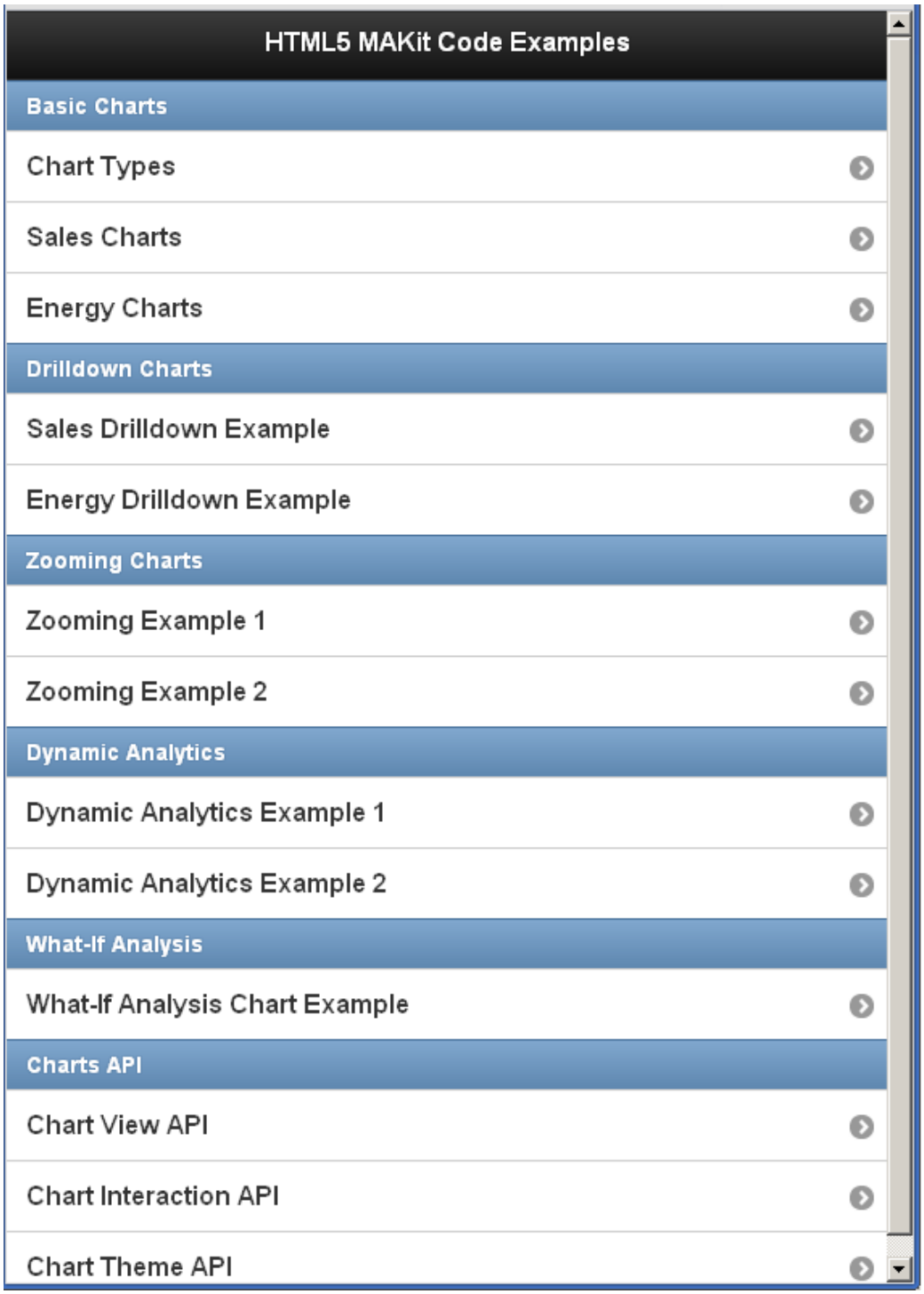
e. In Project Explorer, expand the `WebContent` folder, right-click **`index.html`** and select **Run As** > **Run on Server.**

f. Click **Choose an existing server,** select **HTTP Preview at localhost** and click **Next.** Click **Finish** in the next screen.

Wait for the server to start and load `index.html` in the selected external browser.



This completes setting up the code examples in Mobile WorkSpace. From the main menu, select **MAKit Code Examples using jQuery** to see the available code examples.



Code Examples Walkthrough

Basic Charts

This section demonstrates the chart types supported by the MAKit HTML5 library: column (and stacked column), bar, line, pie, and bubble. Each chart is configured in one XML metadata file.

Refer to the `ChartTypes.html` file to see how the MAKit HTML5 JavaScript library is instantiated and loads the metadata file. `ChartTypes.html` loads six charts in normal mode. Each chart is rendered in one element HTML tag.

Examples of Chart Configuration

ChartTypes_Column.xml

```
<Chart Name="ColumnChart" Title="Region/Product Sales" ChartType="column"
Query="SalesOrder">
       <Category Format="" Column="Region" DisplayName="Region" />
       <Series Format="" Column="Product" DisplayName="Product" />
       <Values>
              <Value Format="currency" Expression="SalesAmount" />
       </Values>
 </Chart>
```

ChartTypes_Bar.xml

```
<Chart Name="BarChart" Title="Salary By Dept/Gender" ChartType="bar" Query="DeptSalary">
       <Category Format="" Column="Department" DisplayName="Department"  />
       <Series Format="" Column="Sex" DisplayName="Sex" />
       <Values>
              <Value Format="round2" Expression="Salary" DisplayName="Salary By Gender" />
       </Values>
 </Chart>
```

ChartTypes_Pie.xml

```
<Chart Name="PieChart" Title="Printer Sales " ChartType="pie" Query="PrinterSales">
       <Category Format="" Column="SalesRep" DisplayName="Sales Rep"  />
       <Values>
              <Value Format="rounded0" Expression="Sales" DisplayName="Units Sold" />
       </Values>
 </Chart>
```

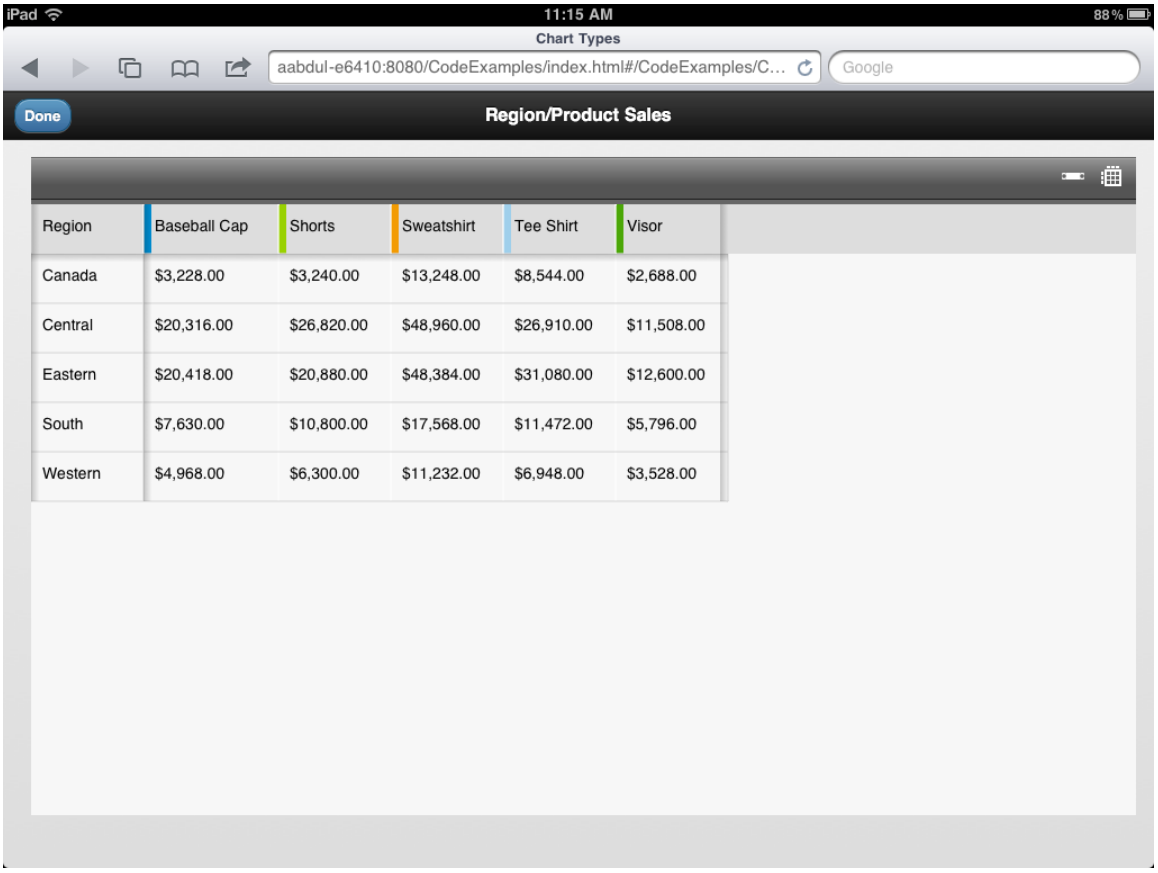
ChartTypes_Bubble.xml

```
<Chart Name="BubbleChart" Title="US Crime Data" ChartType="bubble" Query="CrimeData">
       <Category Format="" Column="State" DisplayName="US State" />
       <Series Format="" Column="CrimeType" DisplayName="CrimeType" />
       <Values>
              <Value Format="rounded2" Expression="Ratio" DisplayName="Crime Ratio" />
       </Values>
</Chart>
```

The charts render in normal mode in supported mobile browsers (or Safari on an iPad):

Click the arrow in the upper-right corner of any chart to load it in full-view mode:

Click the **Table** button to show data table for the chart.



Region	Baseball Cap	Shorts	Sweatshirt	Tee Shirt	Visor
Canada	$3,228.00	$3,240.00	$13,248.00	$8,544.00	$2,688.00
Central	$20,316.00	$26,820.00	$48,960.00	$26,910.00	$11,508.00
Eastern	$20,418.00	$20,880.00	$48,384.00	$31,080.00	$12,600.00
South	$7,630.00	$10,800.00	$17,568.00	$11,472.00	$5,796.00
Western	$4,968.00	$6,300.00	$11,232.00	$6,948.00	$3,528.00

Click **Table** again to return the chart to full-view mode.

Drilldown Charts

MAKit HTML5 lets you drill down by category, or by category and series, on a single chart. You can define any level of drilldown chart in the metadata definition.

You can drill down only when the chart is in full-view mode, by double-tapping a selected category (ByCategory) or by double-tapping individual series in a chart.

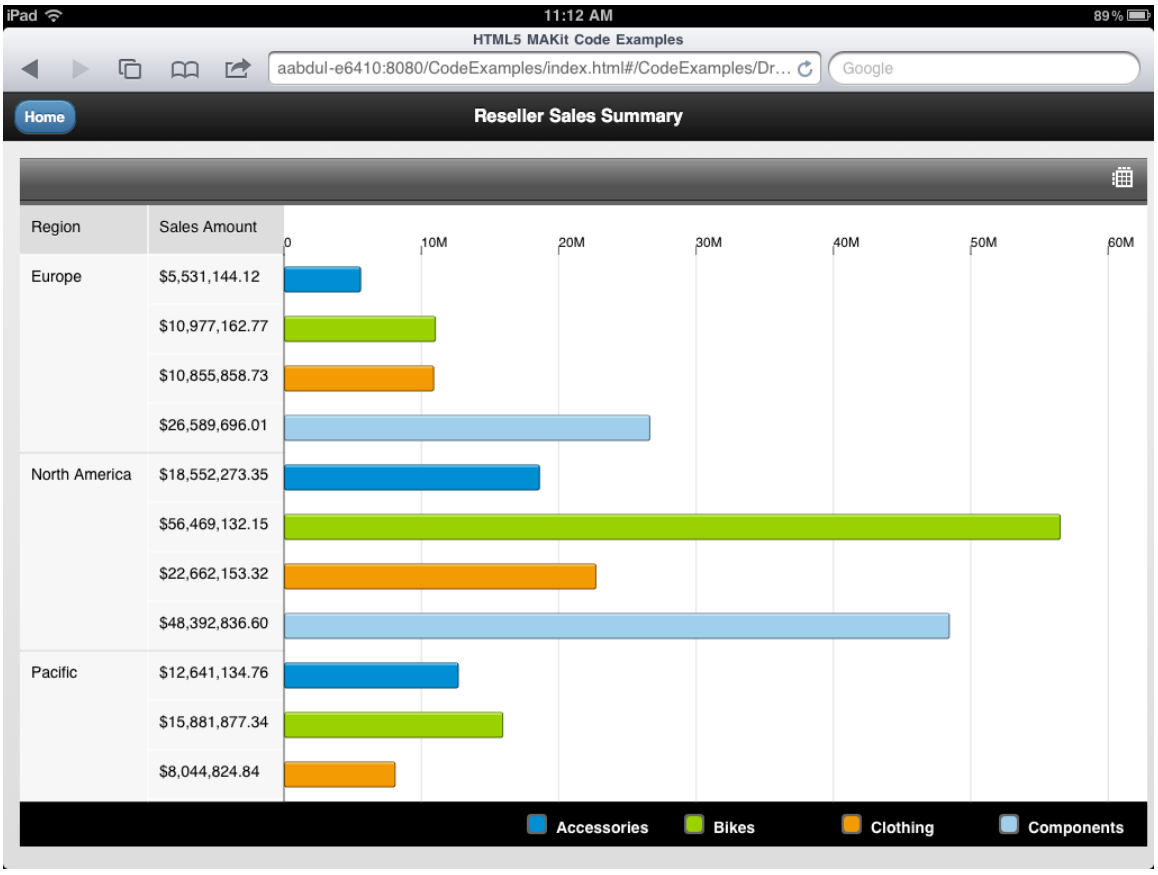
Configure the drilldown charts using `<DrillDownGroup>….</DrillDownGroup>` tags; configure individual charts at each level using `<Chart> …. </Chart>` tags.

See the `Drilldown_Resellers.xml` and `Drilldown_OilPrices.xml` metadata files to see the drilldown configurations.

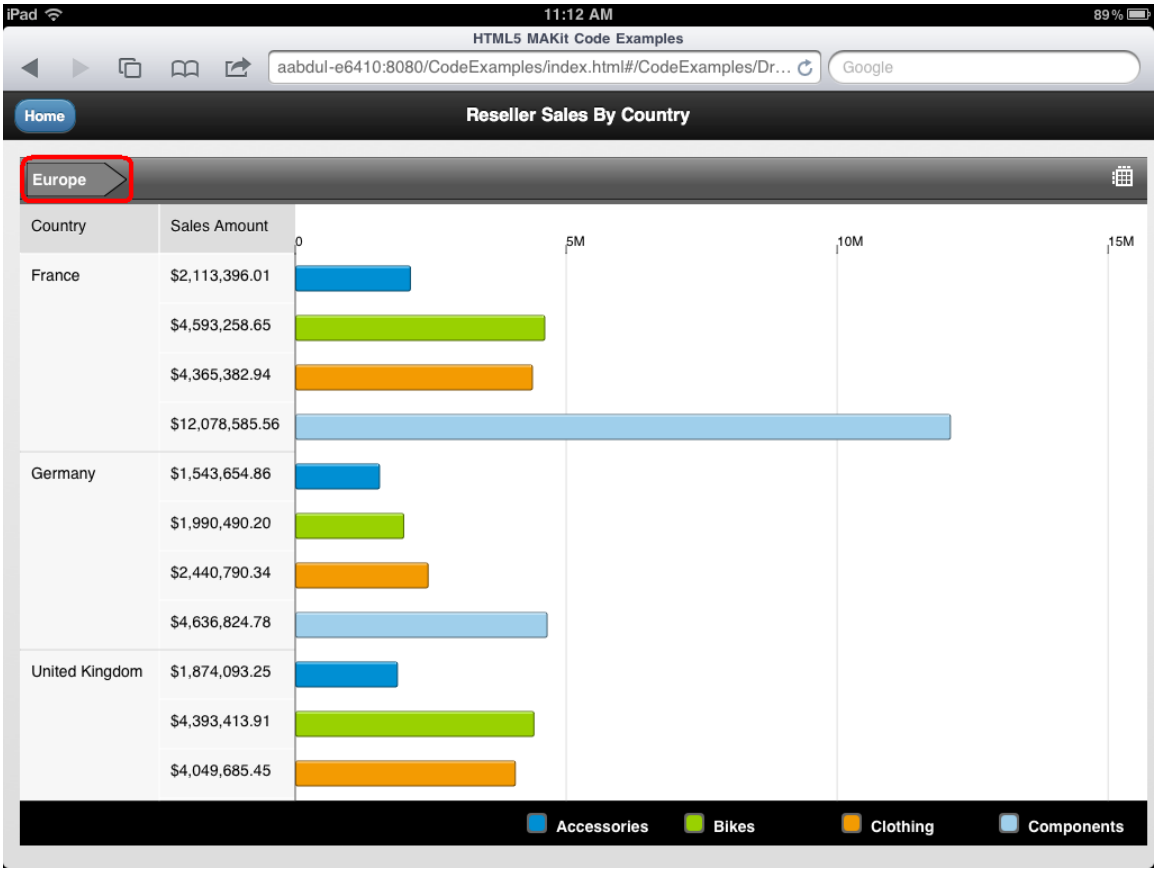
Example of DrillDown Configuration

```
<DrillDownGroup>
  <Chart Name="ResellerSales" Title="Reseller Sales Summary " ChartType="bar"
                      Query="ResellerSummary" >
      <Category Format="" Column="Region" DisplayName="Region" />
      <Series Format="" Column="ProductCategory" DisplayName="ProductCategory" />
      <Values>
         <Value Format="currency" Expression="SalesAmount"  DisplayName="Sales Amount" />
      </Values>
  </Chart>
  <Chart Name="SalesByCountry" Title="Reseller Sales By Country" ChartType="bar"
                                 Query="ResellerByCountry" >
     <Category Format="" Column="Country" DisplayName="Country" />
     <Series Format="" Column="ProductCategory" DisplayName="Product Category" />
     <Values>
        <Value Format="currency" Expression="SalesAmount"  DisplayName="Sales Amount" />
     </Values>
  </Chart>
  <Chart Name="SalesByCategory" Title="Sales By Product Category " ChartType="bar"
                          Query="ResellerByCategory">
     <Category Format="" Column="ProductSubCategory" DisplayName="Product Sub Category" />
     <Values>
         <Value Format="currency" Expression="SalesAmount"  DisplayName="Sales Amount" />
      </Values>
   </Chart>
</DrillDownGroup>
```

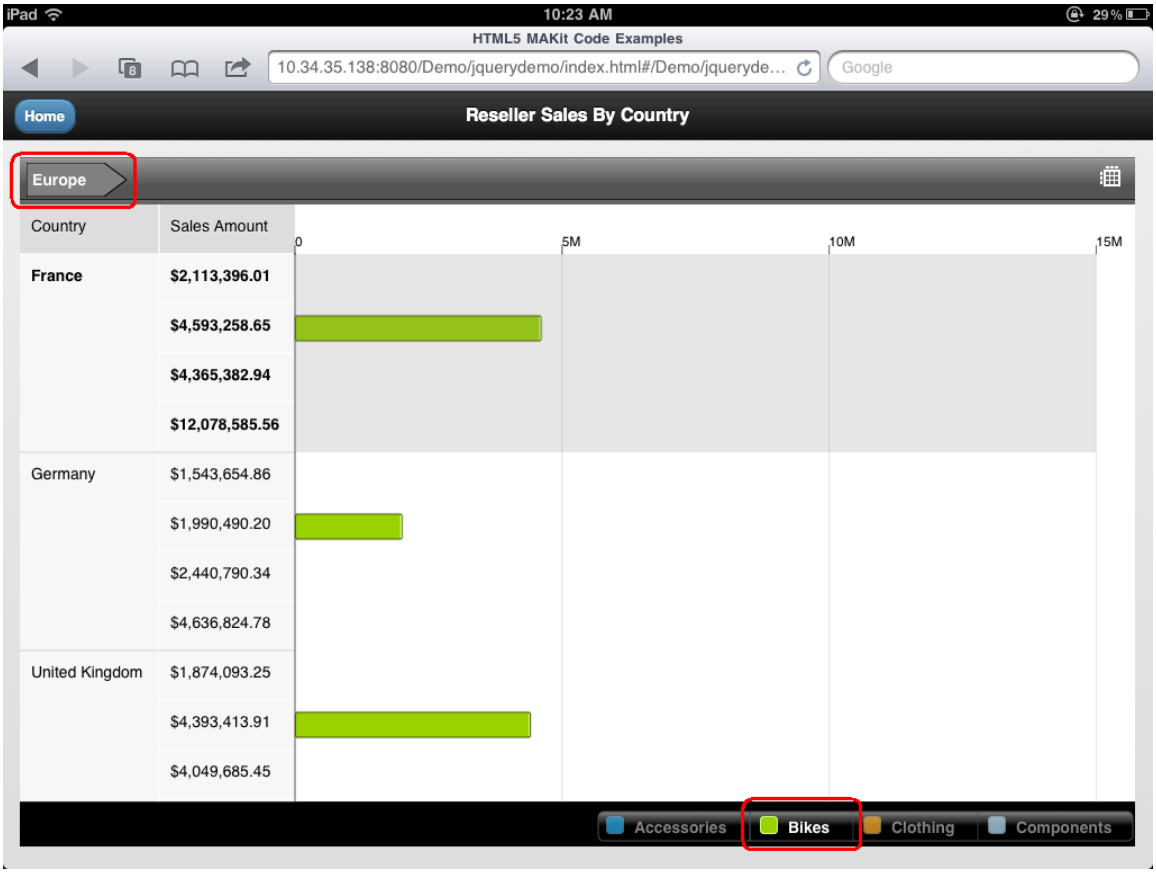
Drilldown charts render similarly to other charts in their initial view:



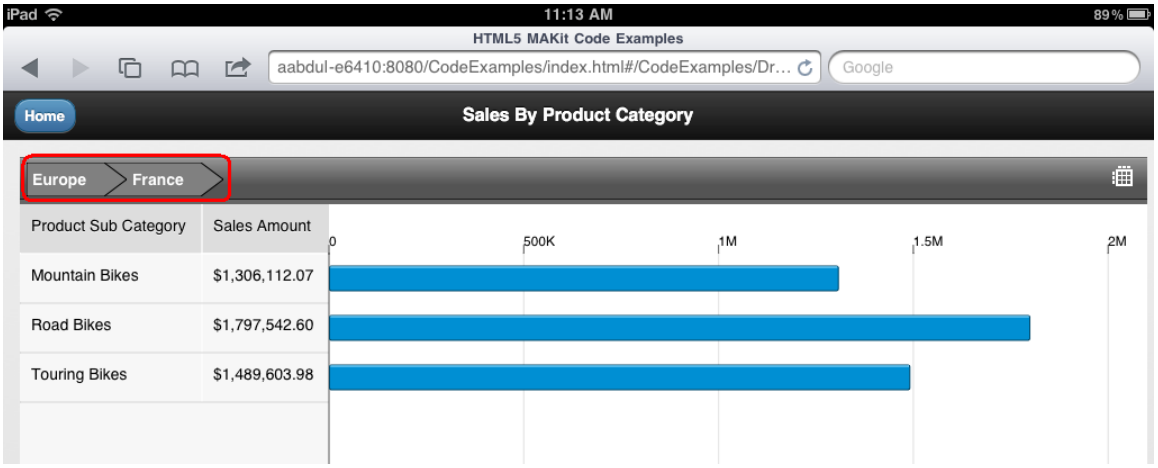
Double-tap in the **Europe** region to drill down by category:



Select **Bikes** from the legend, then double-tap in the **France** region:



The chart listing the sales of **Bike Types** in **France:**



Zooming Charts

Semantic zooming lets users view a large data set in different scales. Users can see an overview of a large chart by zooming out, and get every detail by zooming in.

Configure zooming charts and chart behavior at each zooming level in the metadata configuration file.

Like drilldown charts, zooming can be performed only in full-view mode, via hierarchy buttons.

Example of Zooming Chart Configuration

```
<Chart Name="ZoomingChart" Title="Sales By Year" ChartType="Line" Query="ResellerYear" >
    <Category Format="" Column="SalesYear" DisplayName="Sales ByYear-Quarter-Month" />
    <Series Format="" Column="AmountType" DisplayName="Cost Type" />
    <Values>
        <Value Format="Rounded2" Expression="Amount"  DisplayName="Sales Amount" />
     </Values>
     <!-- Zooming Levels Configuration -->
     <ZoomingLevels>
        <ZoomingLevel Query="ResellerYear" CategoryColumn="SalesYear" Title="Year" />
        <ZoomingLevel Query="ResellerQtr" CategoryColumn="Quarter" Title="Quarter" />
        <ZoomingLevel Query="ResellerMonth" CategoryColumn="Month" Title="Month" />
    </ZoomingLevels>
 </Chart>
```

The chart initially renders at the first zooming level configured in the metadata (in this example, Sales by Year), and adds hierarchy buttons—one for each level ("Year," "Quarter," and "Month").

Chart at ZoomLevel=1:

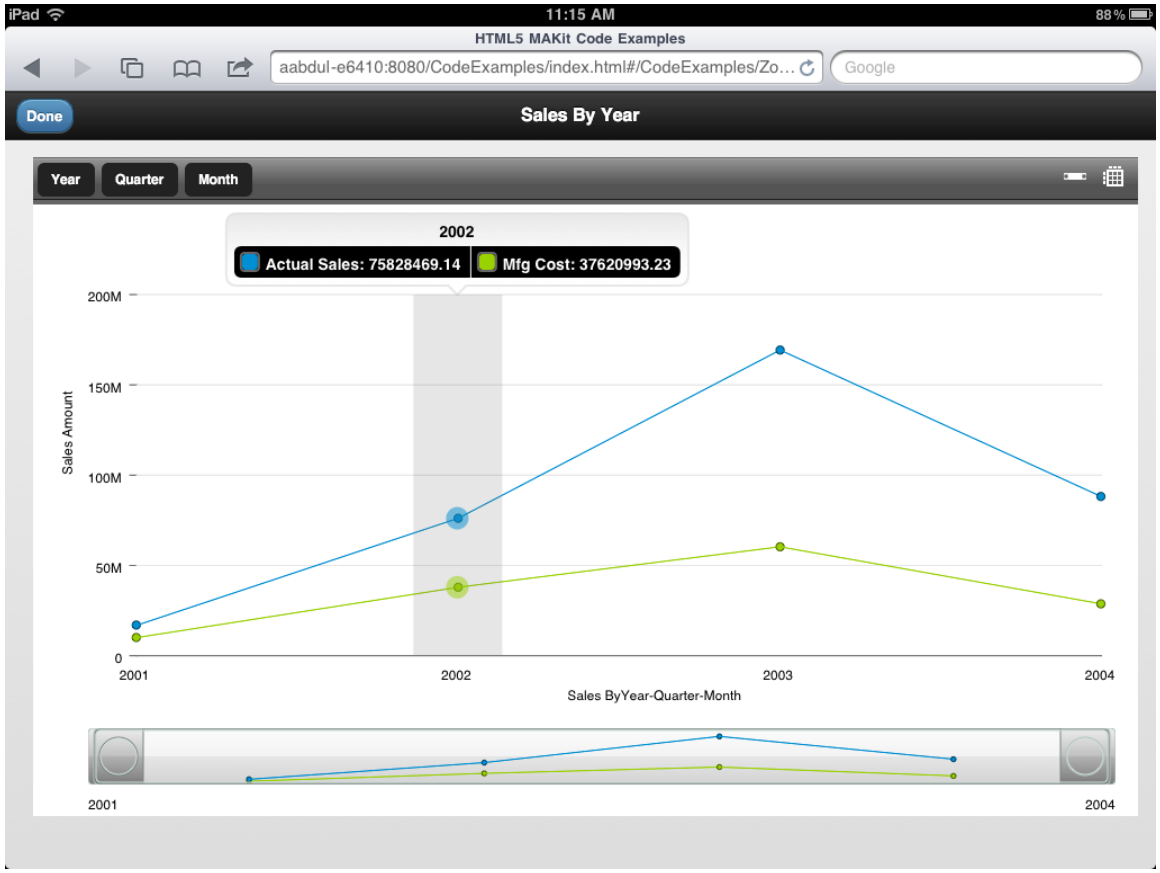


Chart at ZoomLevel=2:

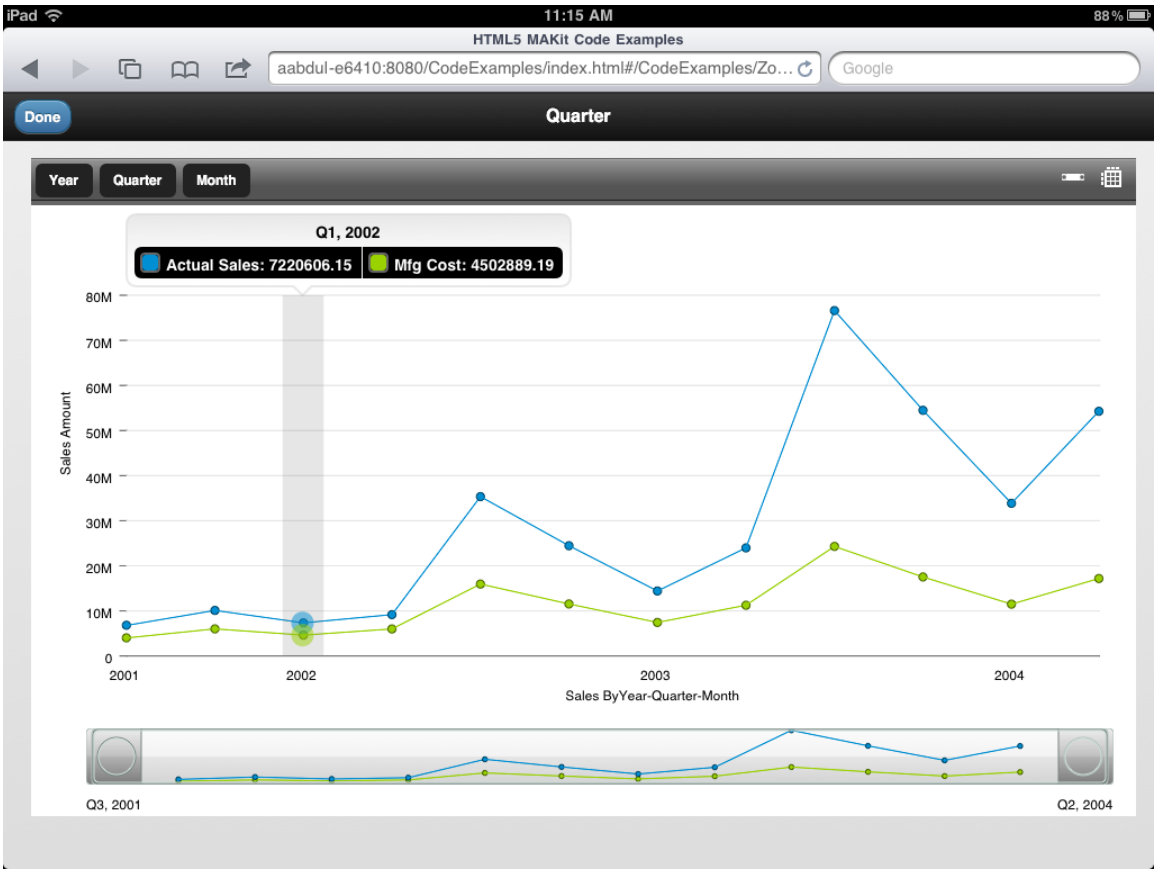
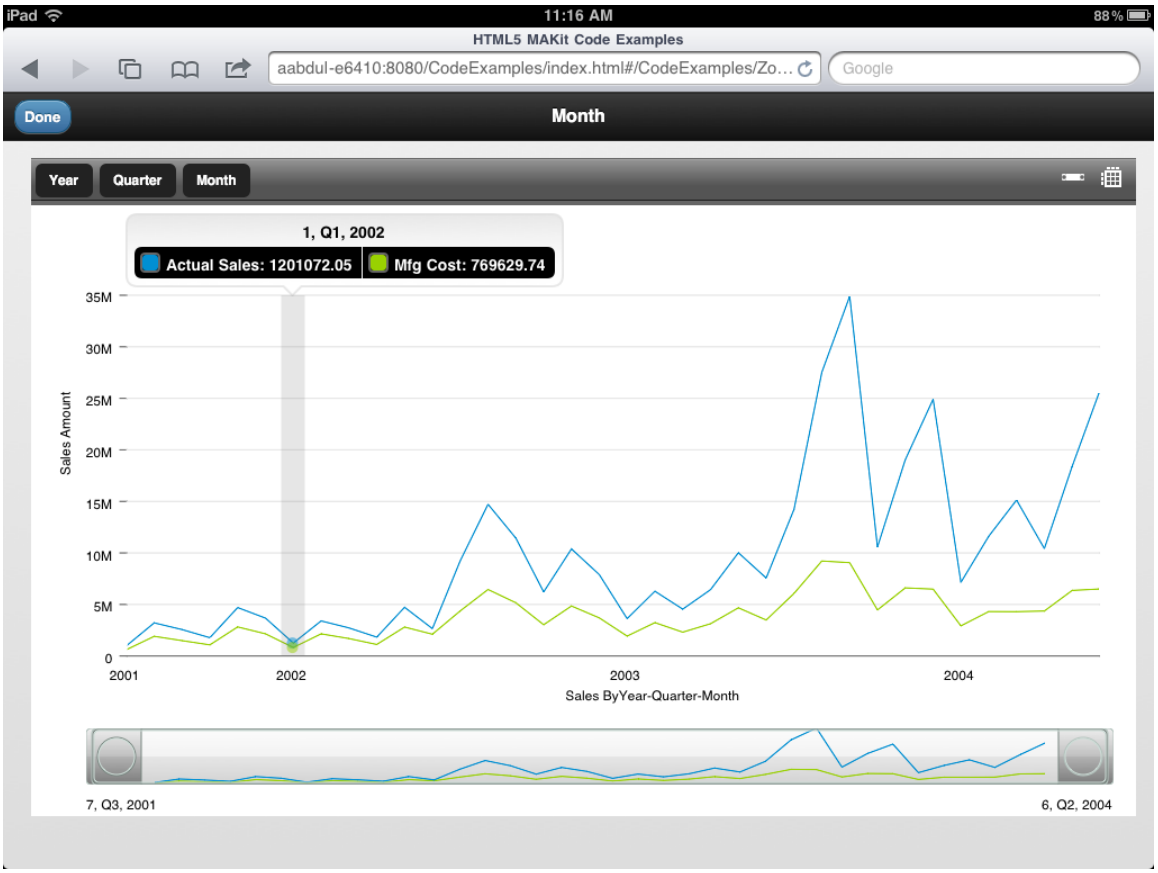
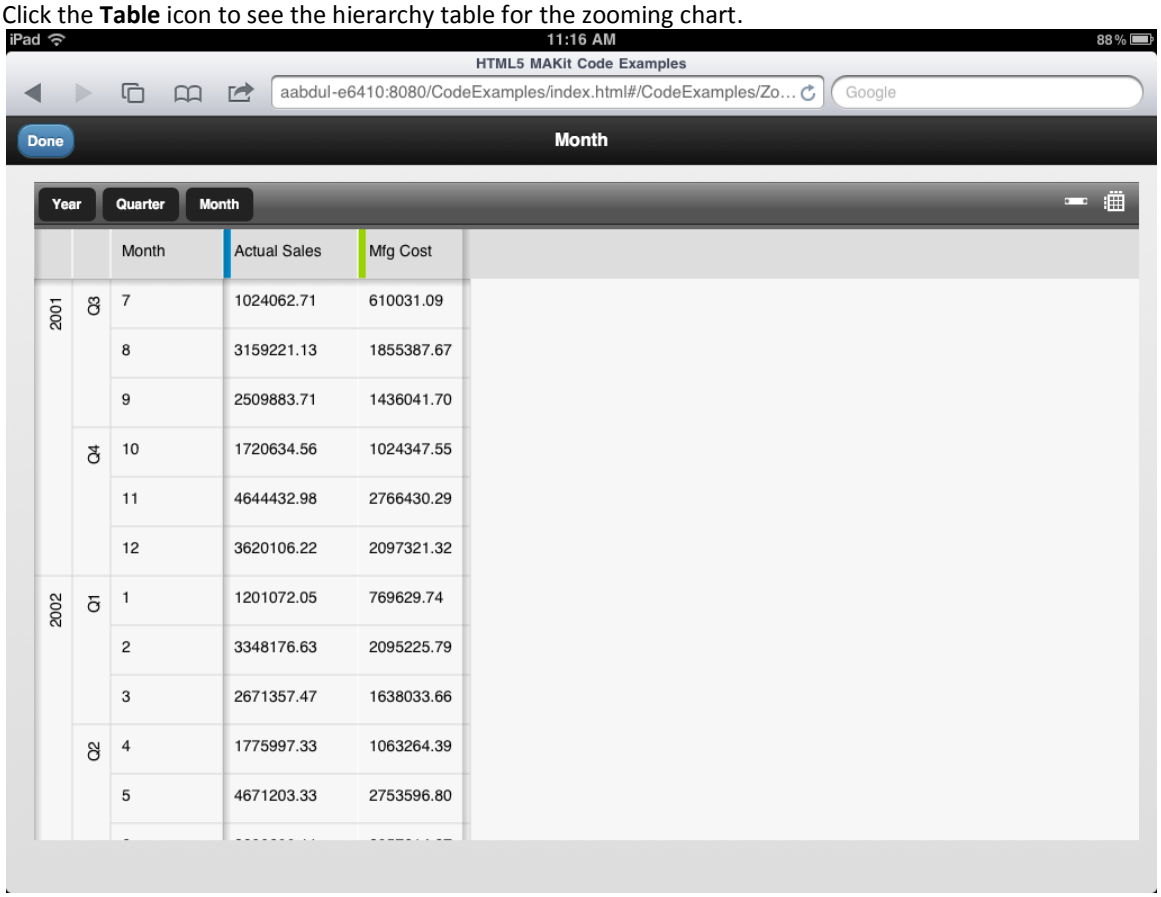


Chart at ZoomLevel=3:



Click the **Table** icon to see the hierarchy table for the zooming chart.



		Month	Actual Sales	Mfg Cost
2001	Q3	7	1024062.71	610031.09
		8	3159221.13	1855387.67
		9	2509883.71	1436041.70
	Q4	10	1720634.56	1024347.55
		11	4644432.98	2766430.29
		12	3620106.22	2097321.32
2002	Q1	1	1201072.05	769629.74
		2	3348176.63	2095225.79
		3	2671357.47	1638033.66
	Q2	4	1775997.33	1063264.39
		5	4671203.33	2753596.80

Dynamic Analytic Panel

This section demonstrates the dynamic analytic capabilities of MAKit. Dynamic analytics is a powerful feature that lets you interactively visualize reports by changing some parameters at runtime, and refreshing charts based on the new values.

The heart of dynamic analytics is the variable definition in the metadata, which constructs the value axis expressions using the variables. When the device user changes the variable value at runtime, MAKit evaluates the value axis expression, and refreshes the charts based on the new value.

Define variables in a separate XML metadata file using `<Variables>……</Variables>` tags.

MAKit automatically generates UI controls to manipulate the values of these variables:

Variable Type	User Interface Generated by MAKit
integer	Slider control
float	PickerView control
boolean	Switch control
enum	PickerView control
datetime	Date Picker control

Example of Variable Definition

DA_Sales_Variables.xml

```
<Variables>
   <Variable Name="$salesdrive" Description="Is Global Sales Promotion ON ?"
                             VariableType="boolean" Value="0" />
   <Variable Name="$region" Description="Sales Promotion in Region "
                     VariableType="enum" Selection="1">
        <Choices>
            <Choice Display="N/A" Value="n/a" />
            <Choice Display="Eastern" Value="Eastern" />
            <Choice Display="Western" Value="Western" />
            <Choice Display="Central" Value="Central" />
            <Choice Display="South" Value="South" />
            <Choice Display="Canada" Value="Canada" />
        </Choices>
    </Variable>
</Variables>
```

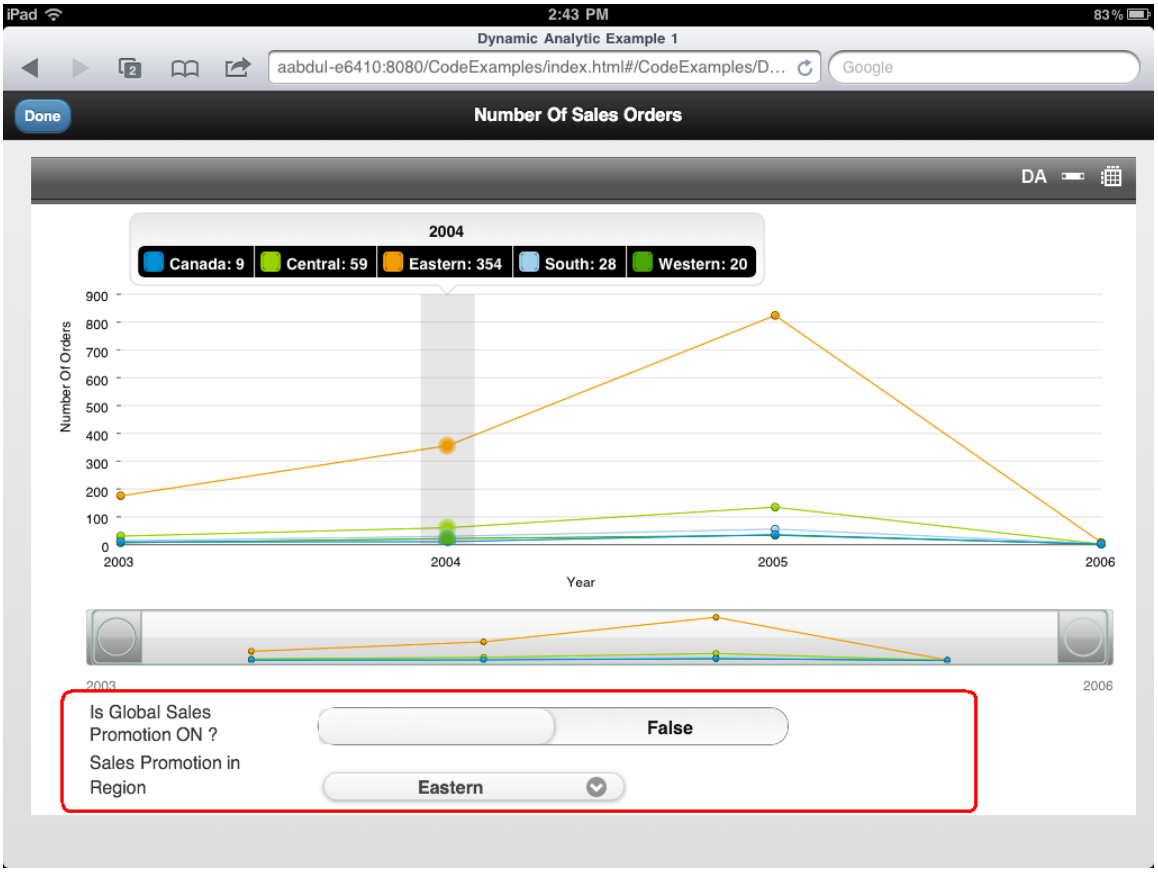
The variables defined in the metadata are used in value expressions of chart configuration as shown here.

DA_SalesPerformance_Chart.xml

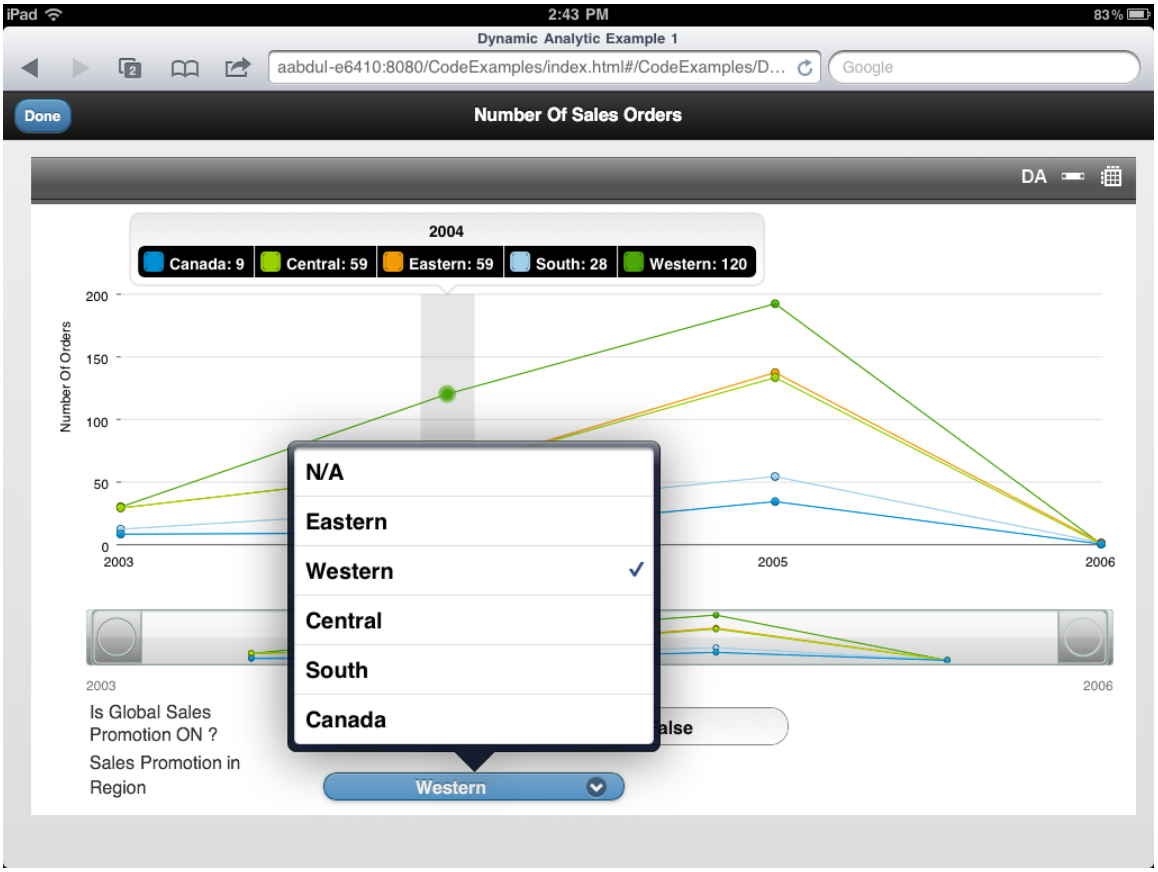
```
<Chart Name="ProductSales" Title="Region/Product Sales" ChartType="column" Query="SalesOrder">
     <Category Format="" Column="Region" DisplayName="Region" />
     <Series Format="" Column="Product" DisplayName="Product" />
     <Values>
         <Value Format="currency" Expression="if($salesdrive,SalesAmount+(SalesAmount *
0.30),if(Region==$region, SalesAmount+(SalesAmount * 0.80),SalesAmount))"
                 DisplayName="Sales Amount" />
     </Values>
</Chart>
```

When variable values are changed, relevant charts are refreshed based on the expression evaluation.

The Dynamic Analytic panel appears when the chart is in full-view mode:



This figure shows the refreshed chart based on a new variable value ("Western"):



What-If Analysis

What-if analysis is similar to dynamic analytics. Both allow the user to change data behind the chart and immediately see the chart updated; both map the data to UI controls on the screen. However, whereas dynamic analytics changes the variables defined in expressions, what-if analysis changes the data retrieved from the datasource.

Configure what-if analysis in the metadata using what-if columns, which are used in value expressions in the chart configuration.

Configure what-if columns using `<WhatIfColumns>…</WhatIfColumns>` tags.

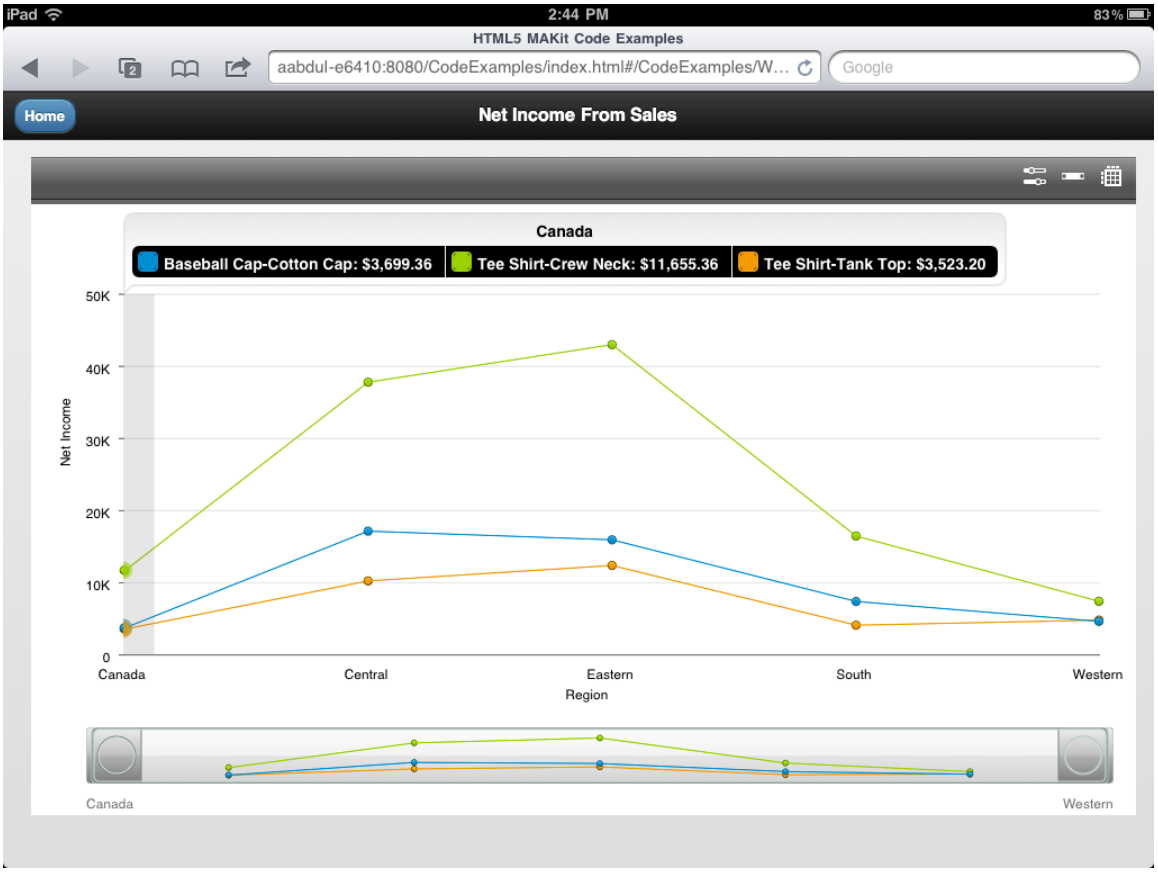
Example of What-If Analysis Configuration

```
<Chart Name="WhatIfChart" Title="Net Income From Sales" ChartType="Line" Query="ProductSales">
   <Category Format="" Column="Region" DisplayName="Region" />
   <Series Format="" Column="Product" DisplayName="Product" />
   <Values>
       <Value Format="Currency" Expression="((UnitPrice * Quantity) - ((Cost_Of_Revenue * Quantity) +
                           (RD_Cost * Quantity) + (Mkt_Cost * Quantity) + (Admin_Cost * Quantity)))"
                          ValueSequence="1" DisplayName="Net Income" />
  </Values>
  <!-- What If Columns -->
  <WhatIfColumns>
     <Column Name="UnitPrice" Description="Unit Price" VariableType="float" Min="1.0" Max="110.0"
                            Step="1.0" Decimals="2" />
     <Column Name="Cost_Of_Revenue" Description="Cost Of Revenue" VariableType="integer" Min="1"
                           Max="40" />
     <Column Name="RD_Cost" Description="R &amp; D Cost" VariableType="integer" Min="1" Max="18"  />
     <Column Name="Mkt_Cost" Description="Marketing Cost" VariableType="float" Min="1.0" Max="10.0"
                            Step="1.0" Decimals="2" />
  </WhatIfColumns>
</Chart>
```

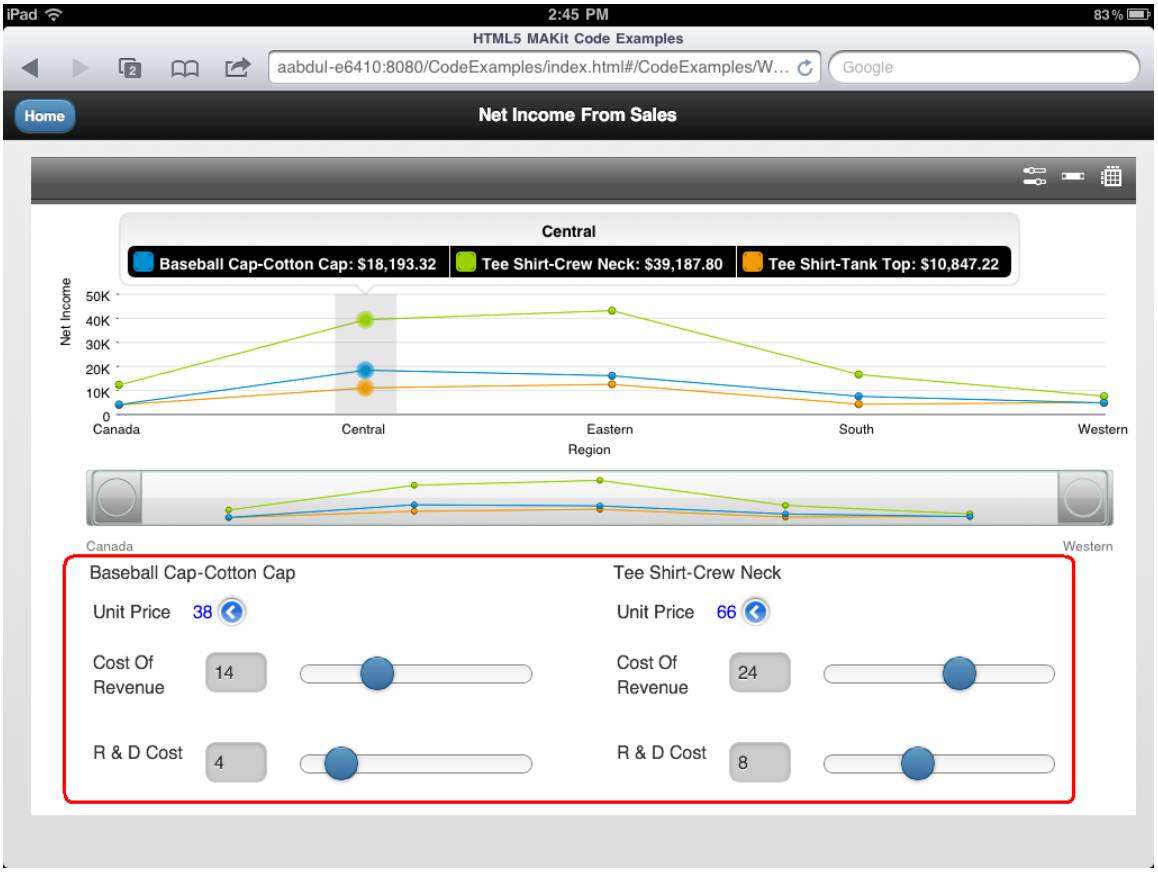
What-if analysis works only in full-view mode, rendering a UI control for each what-if column in the metadata.

To see what-if analysis behavior, select a category and tap the **WI** button in the toolbar to display the What-If panel. Tap the button again to close the What-If panel. When the What-If panel is open, changing the values in a what-if column immediately updates the chart for the selected category.

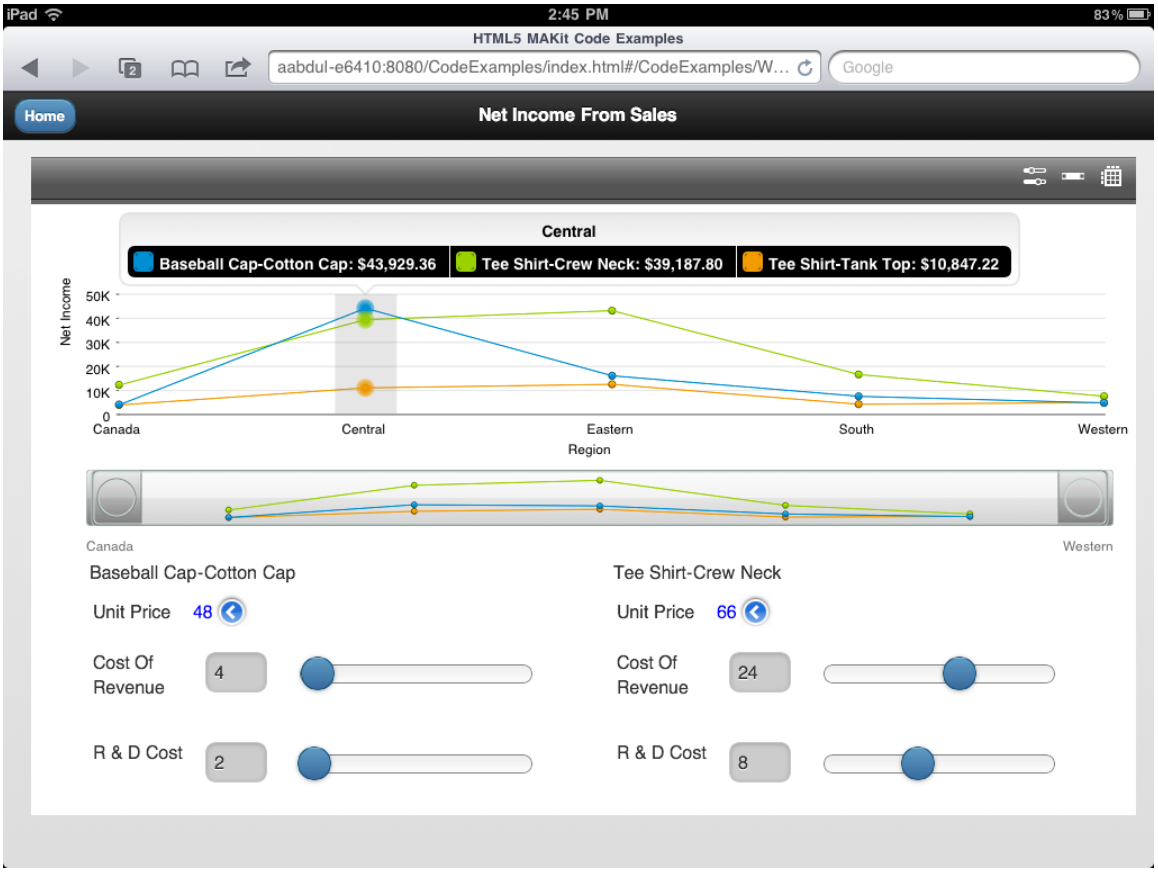
Chart in full-view mode before What-If panel is invoked:



Tap the **WI** button to invoke the What-If panel for the selected category ("Central" region):



Change the values for the What-If column to update the chart accordingly:



MAKit Charts API

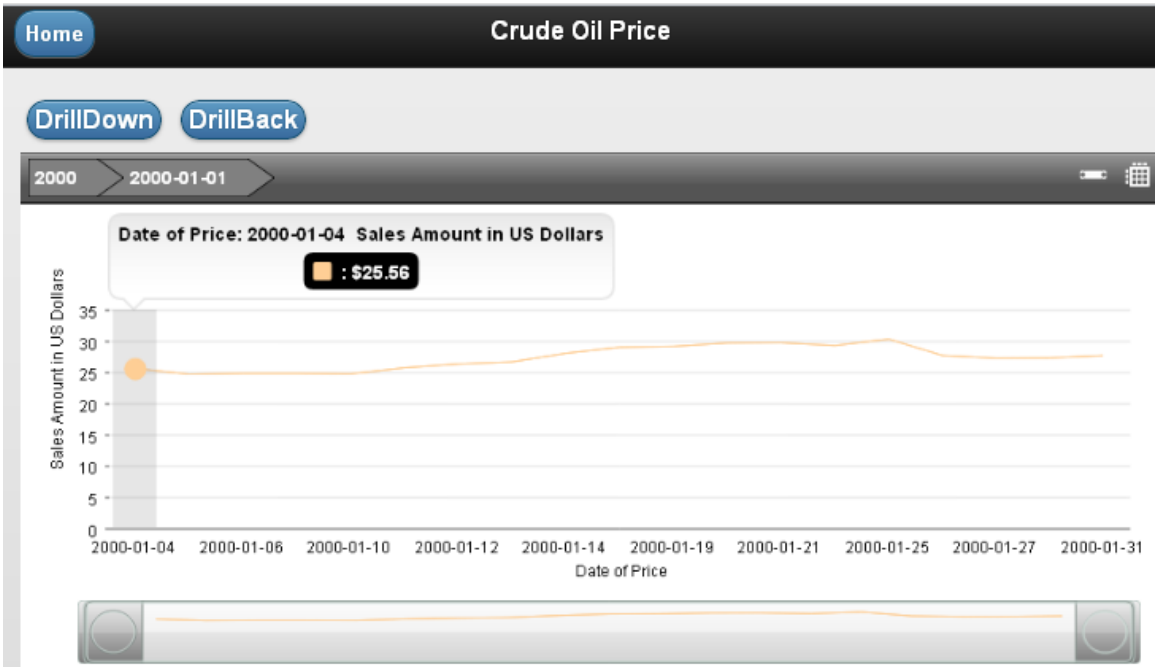
Use the methods in the MAKit Charts API to easily create all chart types. The methods also let you create drilldown levels, select chart themes, show and hide range selectors and toolbars, and change between chart view and table view.

Example of MAKit Charts API Methods Script

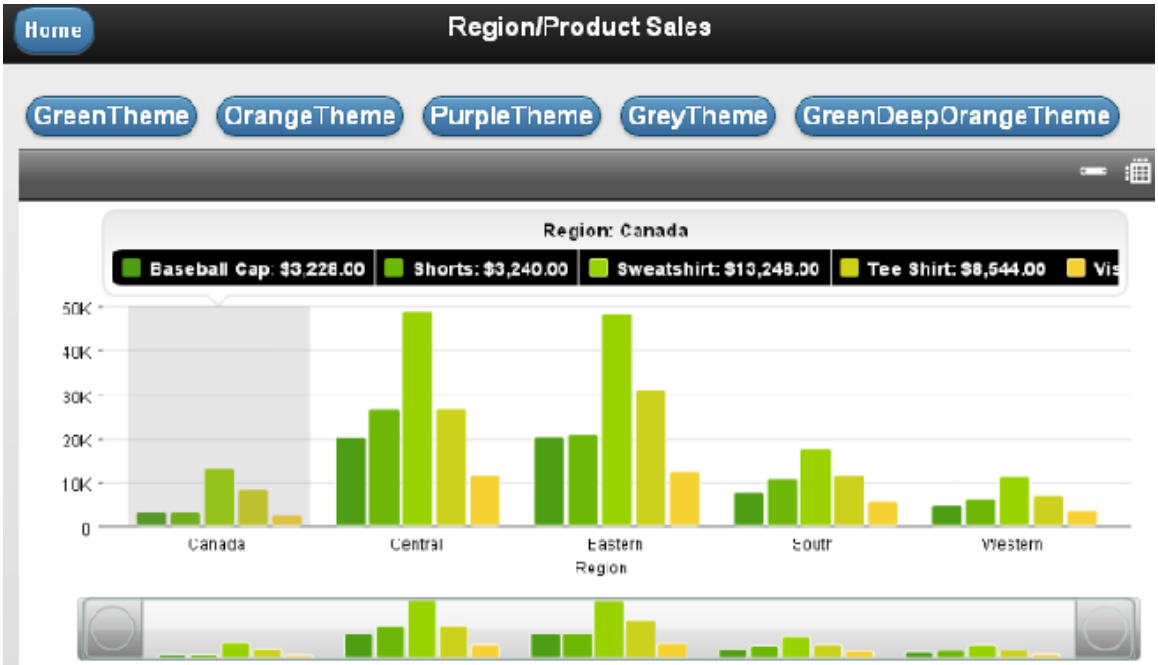
```
$("#rangeselector").click(function() {
    var chart = $MA("fullChart");
    chart.showRangeSelectorView(!chart.isRangeSelectorViewVisible());
    });

$("#toolbar").click(function() {
    var chart= $MA("fullChart");
    chart.showToolBar(!chart.isToolBarVisible());
    });


$("#table").click(function() {
    var chart= $MA("fullChart");
    chart.showTableView(!chart.isTableViewVisible());
    });
```

Use the Charts API methods to show and hide toolbars and range selectors, show a table view, and change between chart types using the same data.

Use MAKit API methods to interact with drilldown charts.



Use the MAKit Theme API to change a chart's color theme.



MAKit 2.2 New Feature

New Chart API Methods

MAKit Charts API methods make it easy to interact with the chart to display the size (area) of a chart, show and hide value bubbles, set the number of visible categories, and so on.

Example of MAKit Charts API Methods Script

- Get Number of Categories

```
$("#category").click(
    function() {
        var elem = document.getElementById("result");
        elem.innerHTML = $MA("fullChart").getNumberOfCategories();
        });
```

- Get Selected Category Index

```
$("#category_index").click(
    function() {
        var elem = document.getElementById("result");
        elem.innerHTML = $MA("fullChart").getSelectedCategoryIndex();
        });
```

- Get Number of Visible Categories

```
$("#category_total").click(function() {
    var elem = document.getElementById("result");
    elem.innerHTML = $MA("fullChart").getNumberOfVisibleCategories();
});
```

- Get Draw Area width

```
$("#area_width").click(function() {
    var elem = document.getElementById("result");
    var DrawArea = $MA("fullChart").getDrawArea();
    elem.innerHTML = "Width = "+ DrawArea.width;
});
```

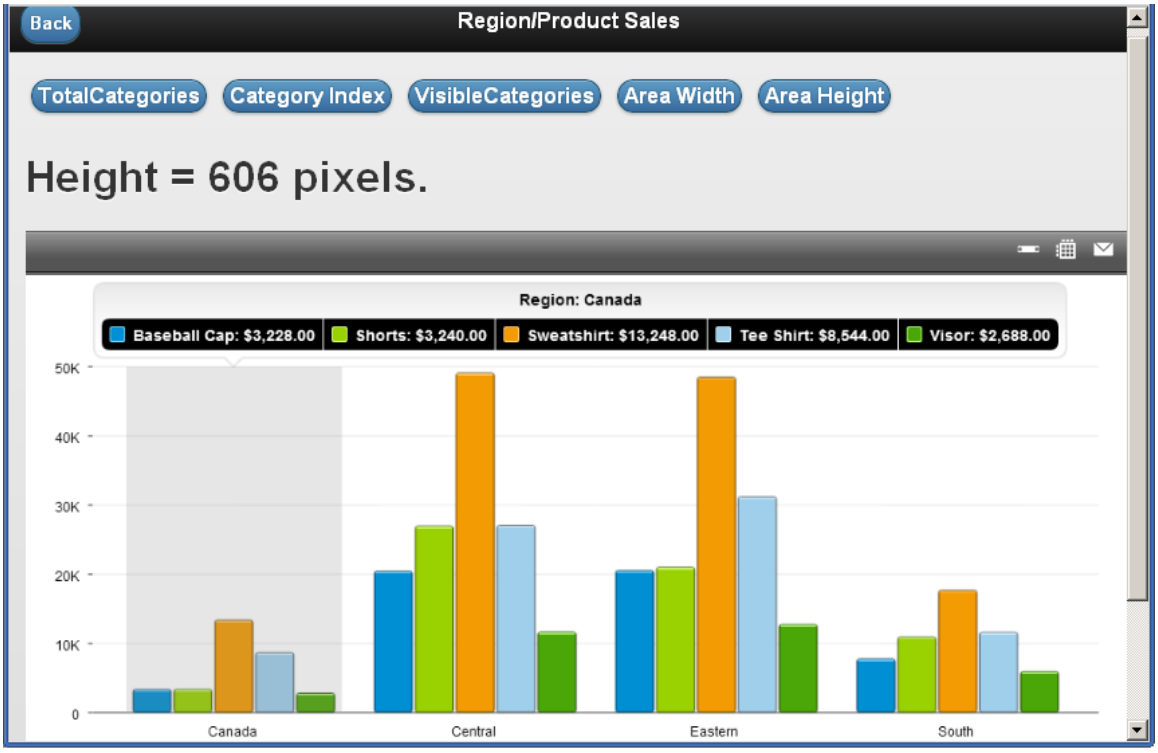
- Get Draw Area Height

```
$("#area_height").click(function() {
    var elem = document.getElementById("result");
    var DrawArea = $MA("fullChart").getDrawArea();
    elem.innerHTML = "Height = "+ DrawArea.height;
});
```

Use the Charts API methods to get the number of visible categories, total categories, index of the category, and the height and width of the drawing area.

This chart uses the Charts API to show the total number of categories in the chart:

This chart uses the Charts API methods to get the drawing area of the chart:



New Events in MAKit 2.2

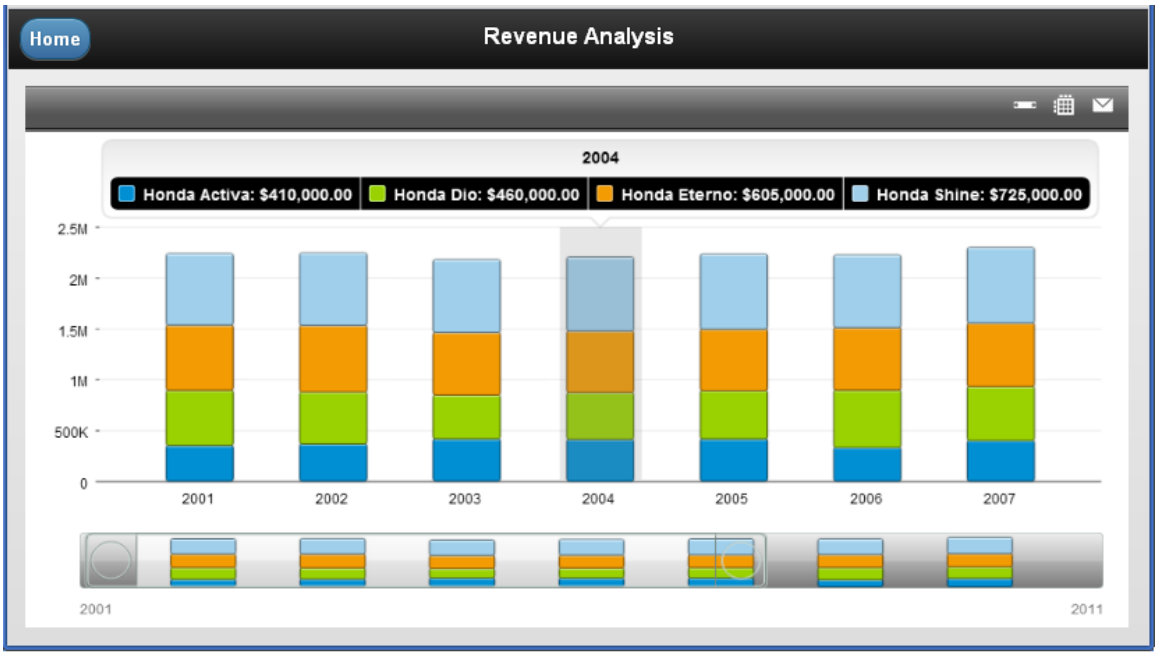
MAKit provides chart touch events that interact with, and receive user actions. You must bind these events before they can be called by scripts.

Example of MAKit Charts Events Script

```
// Bind the Tap Events //
chart.bind("tap", onSingleTap);
chart.bind("doubletap", onDoubleTap);

function onSingleTap() {
    var ls_SelectCat = document.getElementById("result");

    if (ls_SelectCat) {
        ls_SelectCat.innerHTML = "<b>Single Tap Event Triggered, </b>";
        ls_SelectCat.innerHTML += "getSelectedCategory is " +
        ($MA("fullChart").getSelectedCategory());
        }
}

function onDoubleTap() {
    var ls_SelectCat = document.getElementById("result");

    if (ls_SelectCat) {
        ls_SelectCat.innerHTML = "<b>Doublele Tap Event Triggered, </b>";
        ls_SelectCat.innerHTML += "getSelectedSeries is " +
        ($MA("fullChart").getSelectedSeries());
        }
}
```

Use a double-tap event to show the value of a selected series.

This chart uses the **setData** function to load the data from a JSON file type:



Custom Chart Drawings

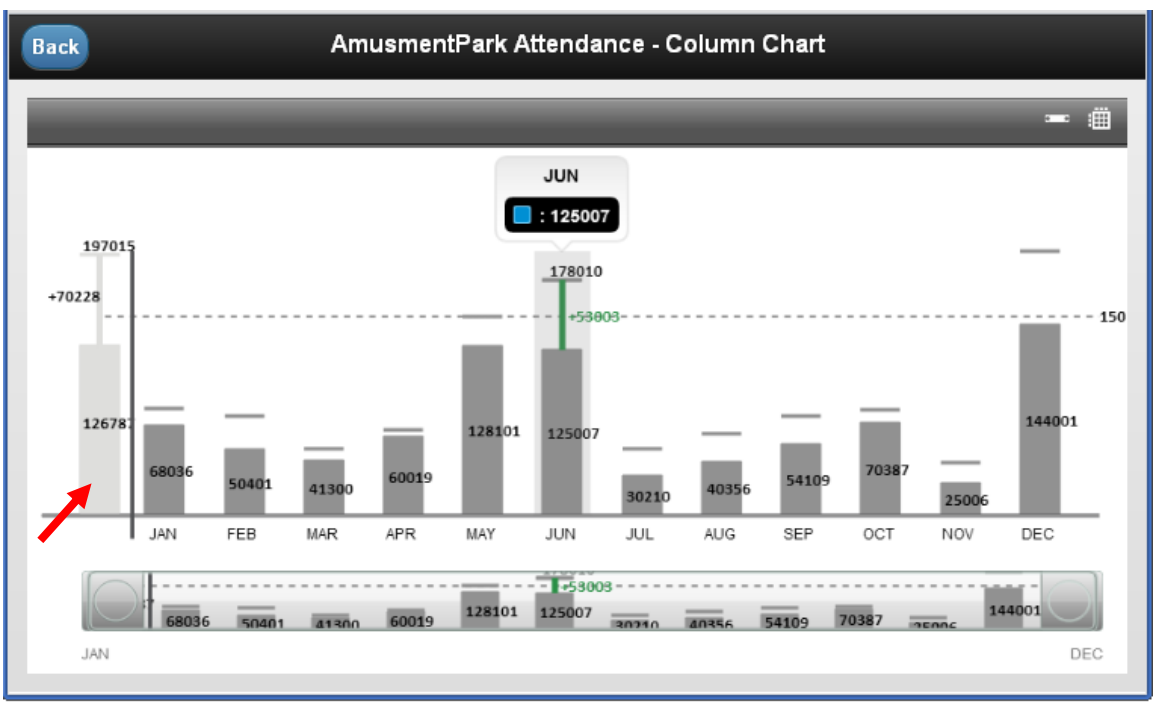
Device users can customize the look and feel of their charts using custom draw events. MAKit custom charts allow a device user to pass custom information, such as dimensions, chart color, item color, and so

on. Device users can also show and hide the axis, axis labels, thickness of the axis, and so on. Custom charts are supported for bar and line charts.

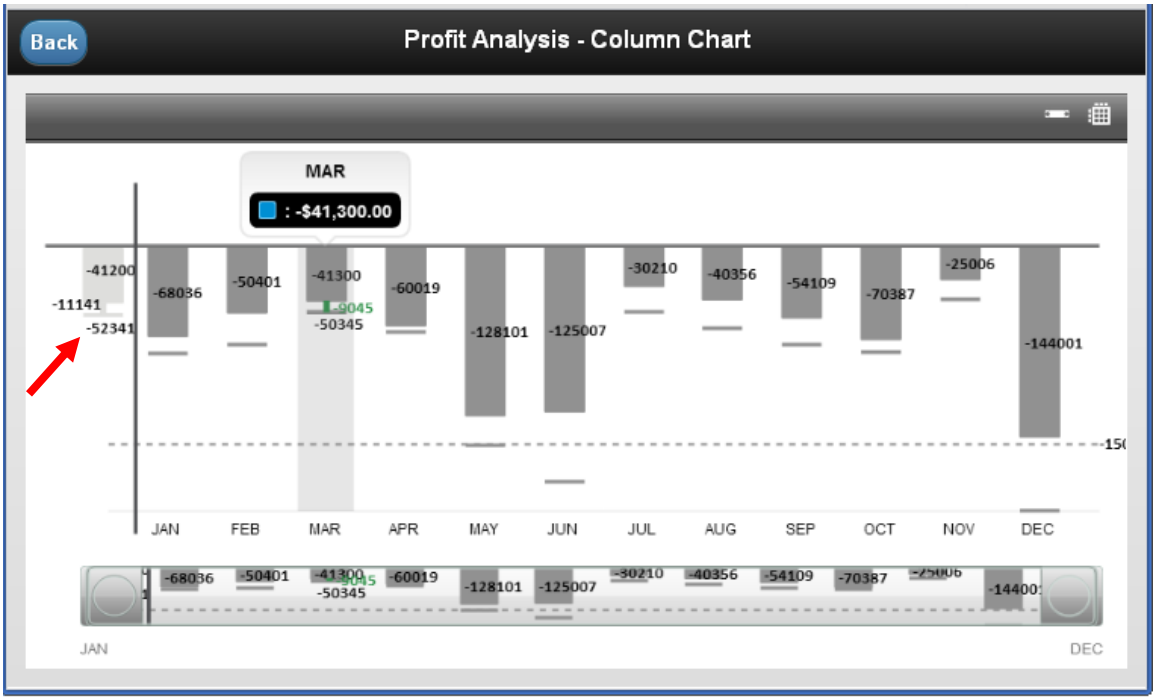
Example of MAKit Custom Chart Script

```
arrange("container",1,1,false);
var chart = new $MA.Chart("chart", true);
chart.bind("draw", onDraw);
chart.bind("tap", onTap);
chart.bind("initialized", onInitialized);
chart.bind("titlechanged", onTitleChanged);
chart.load("metadata/charttype/ChartTypes_SPColumn.xml");

function onInitialized(id) {
    var chart = $MA(id);
    chart.preventDefaultRendering(true);
    chart.setProperty("values.showlabel", true);
    chart.setProperty("values.showgrid", true);
    chart.setProperty("values.min", 0);
    chart.setProperty("values.max", 200000);
}
```

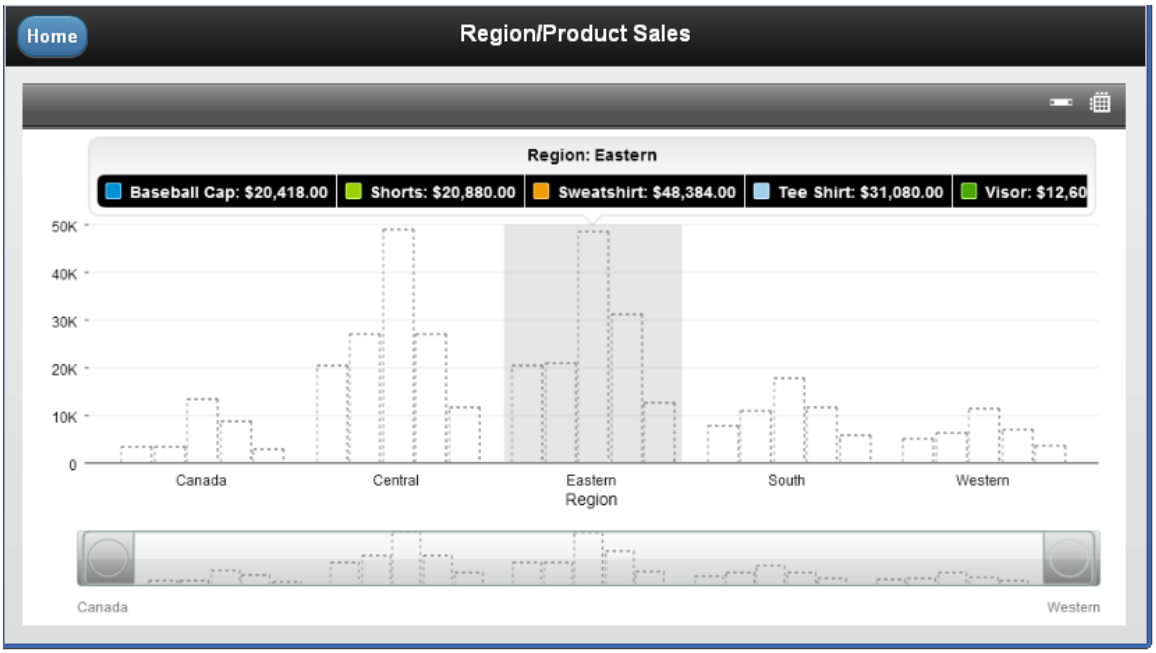
This custom chart shows the previous month value for the selected category:



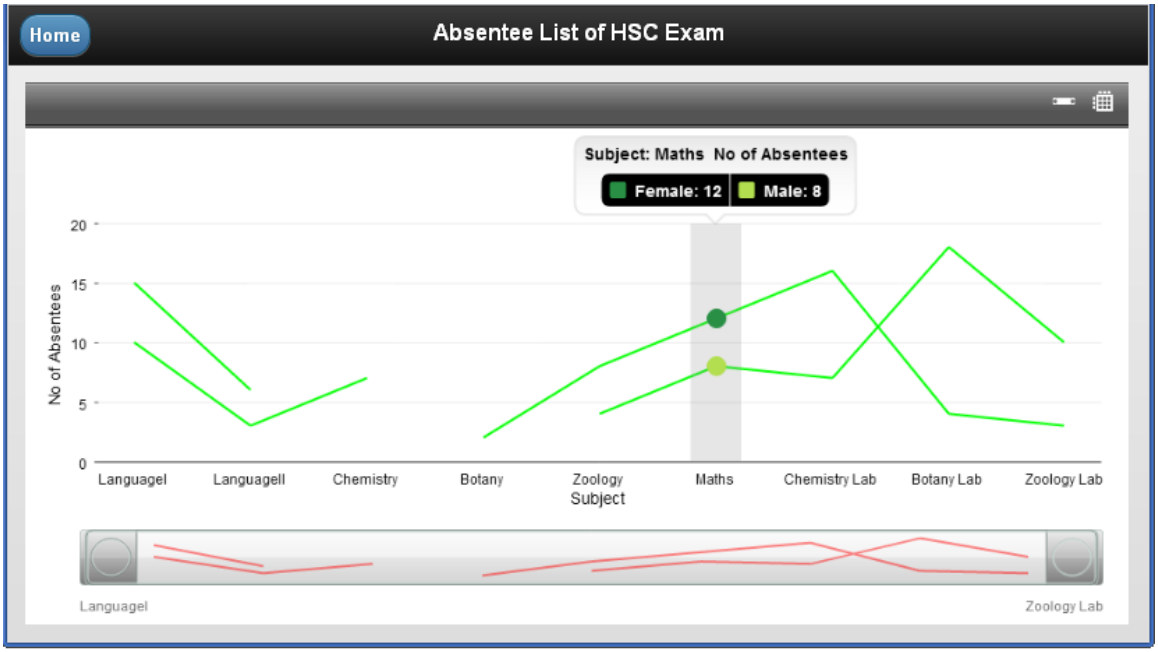
Custom negative chart showing the previous month value of the selected category:



Custom chart rendering dashed bars in regional product sales analysis:

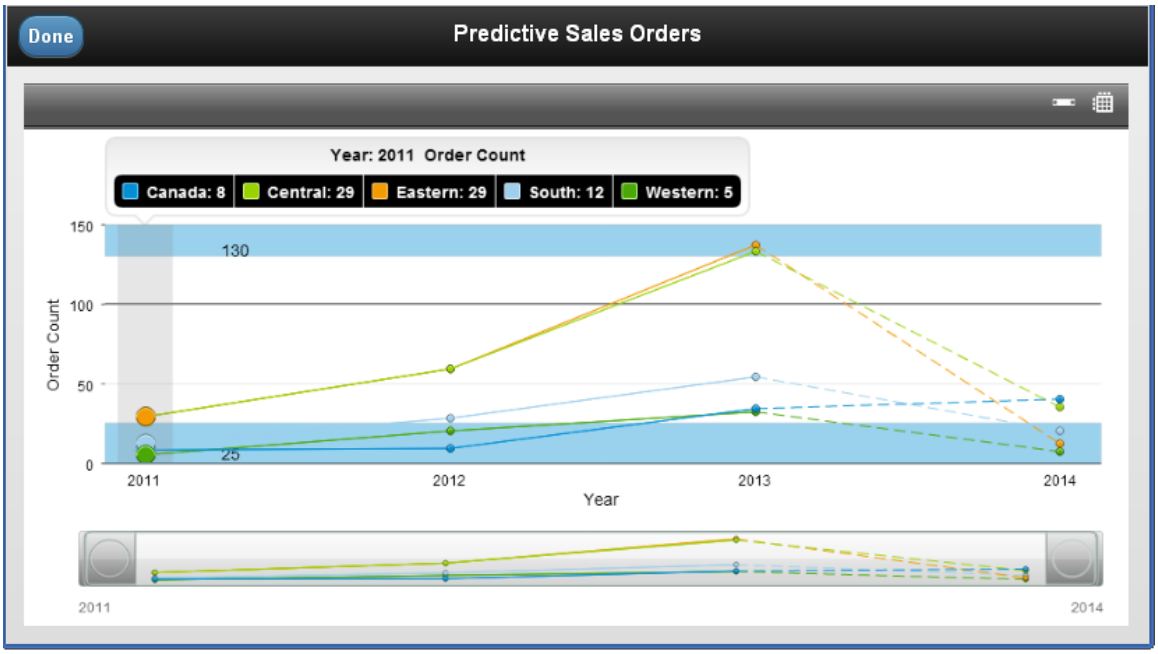


In the line chart, if there is any category with null values, that particular category does not appear. As a result, the line in the chart is not continuous as shown below.



Line Chart Enhancement

In line charts, use the table **setProperty** function to set lower threshold line, upper threshold line, reference, and trend category. The trend category draws dotted lines from the given trend value.



MAKit2.3 New Features

Overlay Charts

Overlay charts let users display two sets of data that share categories as two types of graph in a single chart. Overlay charts plot multiple sets of data on a single chart. Each set of data is based on a different

field, so values are easy to compare. MAKit supports combinations of column and line chart types to present data in different ways on a single chart.

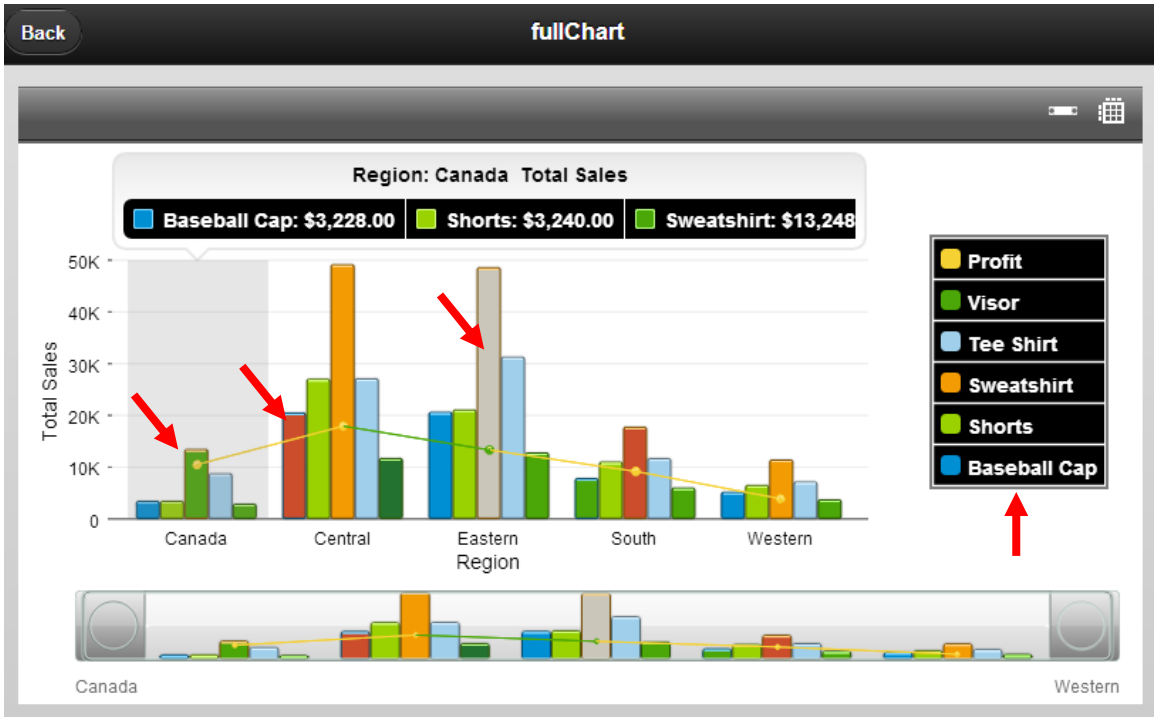
Example of Overlay Chart Configuration

```
<OverlayGroup Name="overlay" Title="Overlay Charts">
        <Category Format="" Column="Region" DisplayName="Region" />

        <Layer Name="Sales" Title="Region/Product Sales Analysis" ChartType="column"
Query="SalesOrder">
                <Series Format="" Column="Product" DisplayName="Product" />
                <Values>
                        <Value Format="currency" Expression="SalesAmount"
DisplayName="Total Sales" />
                </Values>
        </Layer>

        <Layer Name="Profit" Title="Profit Analysis" ChartType="Line"
Query="SalesOrderProfit">
                <Values>
                <Value Format="Rounded2" Expression="Profit" DisplayName="Total Profit" />
                </Values>
        </Layer>
```

Example of MAKit Charts API Methods Script

```
function onInitialized(id)

{

//Set Data Points Color

        $MA(id).setDataPointsColor([[2,"#4ba707"],[5,"#cb4d2c"],[9,"#247230"],[12,"#cac7ba
"],[15,"#4ba707"],[17,"#cb4d2c"]]);


//Set Legend Position

        $MA(id).setLegend("right");

//Hide Value Bubble

        $MA(id).showValueBubble(false);
}


//Single Tap Event

function onSingleTapped(id)

{

        if ($MA(id).isValueBubbleVisible())

        // Hide Value Bubble

        $MA(id).showValueBubble(false);

        else

        // Show Value Bubble

        $MA(id).showValueBubble(true);

}
```

This overlay chart analyzes sales revenue in a column chart and profit in a line chart. The three arrows on the left indicate the colors set by the setDataPointsColor API; the arrow on th eright shows the setLegend.



Bubble Chart Color

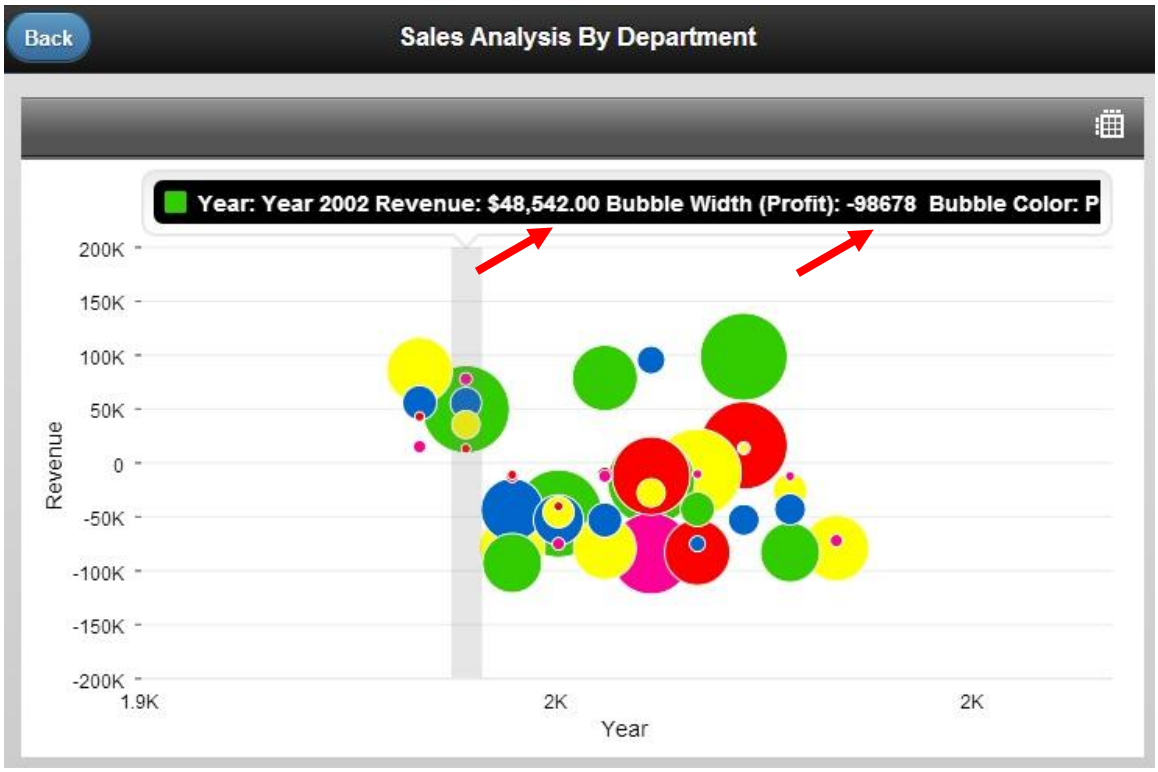
Bubble chart color can represent another measure. The bubble color is similar to the series in other charts—it may represent product names, store names, and so on. The bubble color is determined by the color palette that is defined by the user using the setPalette API. The measure or value that appears in the value bubble may have an additional display value.

Bubble charts support negative values. For example, if a bubble value is positive and a bubble width value is negative, the bubble appears in the positive region; however, the bubble width value appears with a negative sign.

Example of Bubble Chart Configuration

```
<Chart Name="chart 1" ChartType="bubble" Query="BubbleColor" Title="Sales Analysis By
Department">
        <Category Column="year" DisplayText="catdisplaytext" DisplayName="Year" />
        <Values>
                <Value Expression="revenue" Format="currency" DisplayName="Revenue" />
                <Value Expression="width" DisplayName="Bubble Width (Damage)" />
         <Value Expression="color" DisplayText = "coldisplaytext" DisplayName="Bubble
Color (Product)" />
        </Values>
```

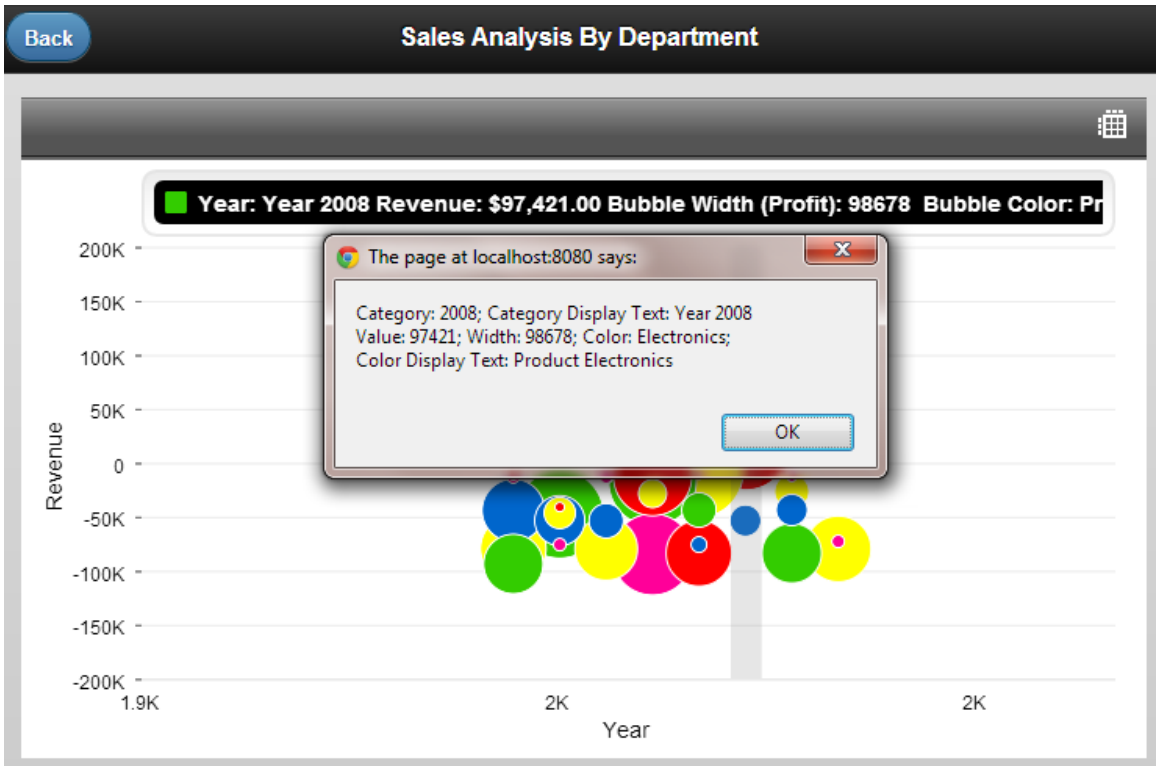
This bubble color chart uses the setPalette API to show the bubbles in custom colors. The explanatory value bubble for the year 2002 includes both a positive value and a negative width.



Example of MAKit Charts API Methods Script

```
//Set Color Palette

var Pallette = [0xFF0000, 0x33CC00, 0xFF0099, 0xFFFF00,  0x0066CC, 0x330033];

function onInitialized(id)

{

        $MA(id).setPalette(Pallette);

}

// Double Tap Event

function onDoubleTap(id)

{

                var chart = $MA(id);

// Get Bubble Information

                var bubbleInfo = chart.getSelectedBubbleInfo();
                var category = bubbleInfo.category;
                var categoryDisplayText = bubbleInfo.categoryDisplayText;
                var value = bubbleInfo.value;
                var width = bubbleInfo.width;
                var color = bubbleInfo.color;
                var colorDisplayText = bubbleInfo.colorDisplayText;

                alert("Category: " + category  + "; Category Display Text: " +
categoryDisplayText + "\nValue: "  + value +
                 "; Width: " + width + "; Color: " + color + "; \nColor Display Text: " +
colorDisplayText);


}
```

This figure shows selected bubble information generated by a double-tap event using the getSelectedBubbleInfo API.



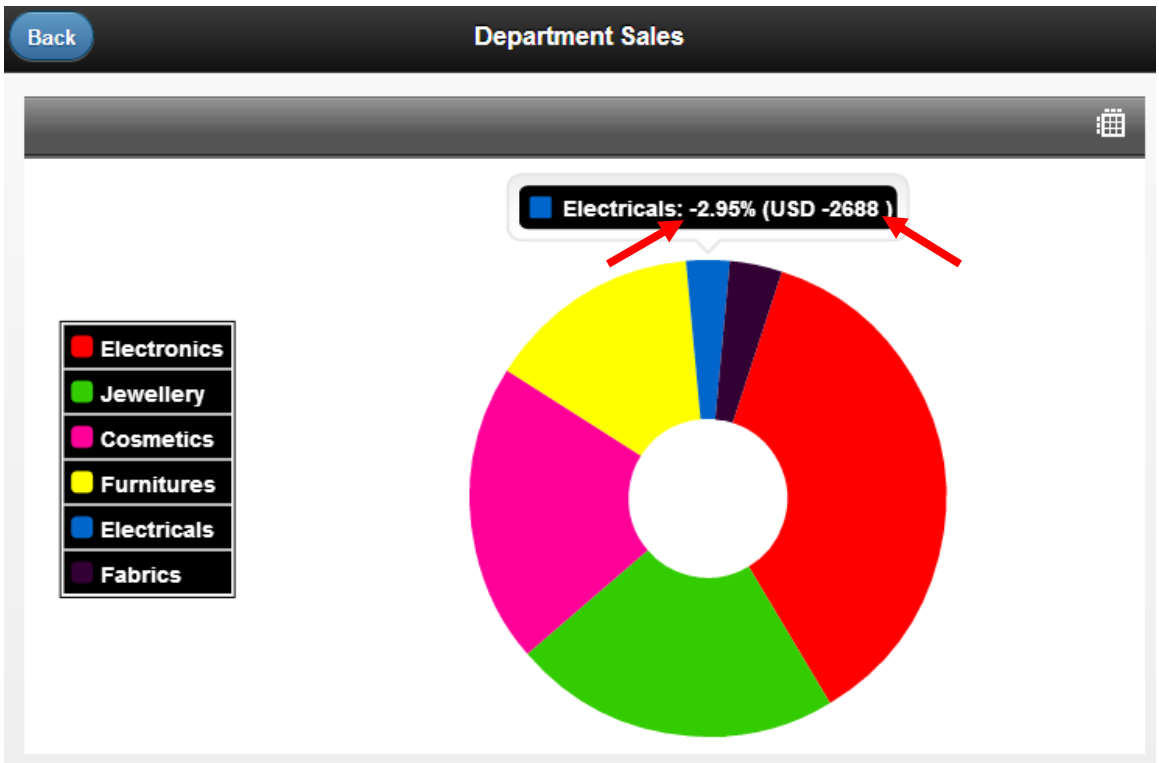
Negative Value Support in Pie Charts

Pie charts now support negative values. Charts are drawn using the absolute value of the data, but the data appears with a negative sign.

Example of Negative Value Pie Chart Configuration

```
<Chart Name="PieChart" Title="Department Sales" ChartType="pie"
Query="SalesOrderNegative">
        <Category Format="" Column="Region" DisplayName="Region" />
        <Values>
                <Value Format="currency" Expression="SalesAmount"
DisplayText="Displaytext" />
        </Values>
```

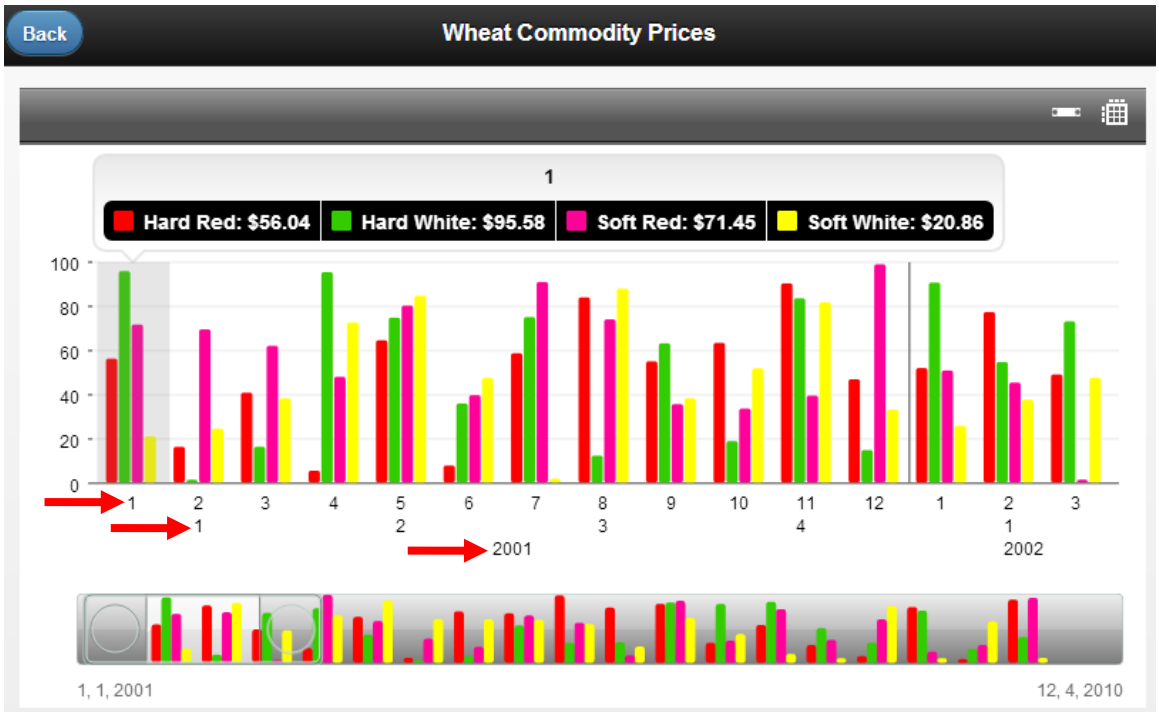
This pie chart shows department sales figures of negative values. The arrow on the left indicates a value with a negative sign, and the arrow on the right indicates the actual percentage value.



Charts with Multiple Categories

MAKit for SAP Mobile Platform 2.3 includes a new property—"Display"—that shows values for all categories in a chart that contains multiple categories. The categories appear in an expanded category axis, and as the last level of value data rendered in the chart.

Example of Multiple Categories Configuration

```
<Chart Name="MultiCategory" Title="Wheat Commodity Prices" ChartType="column"
Query="productrevenuebymonthdata" >
        <Categories display="all">
                <Category Column="year"/>
                <Category Column="quarter"/>
                <Category Column="month" Format="shortmonth"/>
        </Categories>
        <Series Column="product"/>
        <Values>
                <Value Expression="revenue" Format="currency"/>
        </Values>
```

This figure shows multiple values in the category axis. The setPallet API is used to provide the different colors in the columns.