Grails Google Visualization Plugin - Reference Documentation

Benjamin Muschko

Version 2.2-SNAPSHOT

Table of Contents

1. Introduction 1
    1.1. Features 1
    1.2. Issues 1
    1.3. Release Notes 1
    1.4. Acknowledgments 4
    1.5. License 4
2. Usage 5
    2.1. Installation 5
    2.2. Basic usage 5
    2.3. Events 6
    2.4. Formatters 6
3. Charts 8
    3.1. Core Chart 8
    3.2. Additional Charts 10
4. Visualizations 11
    4.1. Available Visualizations 11

Chapter 1. Introduction

This plugin provides a taglib for the interactive charts of the Google Visualization API.

In addition to this document, you may want to read the Google Visualization documentation here.

1.1. Features

- Supports the following visualizations: Annotated Time Line, Area Chart, Bar Chart, Bubble Chart, Candlestick Chart, Column Chart, Combo Chart, Gauge, Geo Chart Geo Map, Intensity Map, Line Chart, Map, Motion Chart, Organizational Chart, Pie Chart, Scatter Chart, Stepped Area Chart, Table Chart, Time Line Chart and Tree Map. See the gallery for more information.
- Implements redesigned charts (Area, Bar, Bubble, Candlestick, Column, Combo, Line, Pie, Scatter and Stepped Area Charts) from the previously known as "Core Chart package" as well as the deprecated versions.
- Provides implementations for table formatters TableArrowFormat, TableBarFormat, TableColorFormat, TableDateFormat, TableNumberFormat and TablePatternFormat.
- Visualization Event Handling.
- Provides support for Data Tables Roles (since version 1.0)

1.2. Issues

The old JIRA links for Grails are not working anymore, so please ignore the issues links in documentation. Also, all issues must be reported to GitHub Issues.

1.3. Release Notes

October 09, 2015 (version 2.0)

- Provides support for Grails 3.x

January 07, 2015 (version 1.0.1)

- Added "title" config option to Calendar chart. Although it does not appear in the API documentation, the examples show that it is available.

January 04, 2015 (version 1.0)

- Updates the configuration options to the current Google Charts API. Important! This might introduce some backward compatibility issues as some parameters have been removed, replaced or have changed type. Eg: tooltipText and tooltipTextStyle replaced by tooltip of type object, ... (This will be noted in the README.md file and the changelog wiki page)
- Removed support for Image Charts as they are no longer part of the Google Charts API
- Now the object type can be declared using the Map notation as the preferred option instead of using an Expando object declaration. It still can be used though to avoid backward compatibility issues. Example: `legend="${[position: 'top', alignment: 'center']}"` instead of: `legend="${new Expando(position: 'top', alignment: 'center')}"`

- Replaced scriptlets with EL expressions in javascript template.
- Fixes issue #15
- Other minor fixes and tasks

December 11, 2014 (version 0.7.2)

- Fixes bug #32: the "error" event handling should be declared before drawing the chart and not after.

November 28, 2014 (version 0.7.1)

- Fixes bugs #28 and #24: Grails 2.3+ XSS prevention mechanism was HTML encoding the javascript output of the taglib and therefore breaking it. There is also a reference in JIRA: [GPGOOGLEVISUALIZATIONAPI-14])

April 30, 2014 (version 0.7)

- Fix ObjectRenderer to properly filter class and metaClass properties (see pull request #23).
- Add support for Timeline and Calendar charts (see pull request #22).

May 9, 2013 (version 0.6.2)

- Enable use of candlestick options (see pull request 12).

February 3, 2013 (version 0.6.1)

- Fixed HTML codec issues (see pull request 11).

November 11, 2012 (version 0.6)

- Internal cleanup (see pull request 10).

October 14, 2012 (version 0.5.6)

- Added missing bar attribute to several charts.

August 18, 2012 (version 0.5.5)

- Support forceIFrame parameter(see GPGOOGLEVISUALIZATIONAPI-12).

August 12, 2012 (version 0.5.4)

- Support vAxes parameter (see GPGOOGLEVISUALIZATIONAPI-11).

July 8, 2012 (version 0.5.3)

- Support for pattern property in NumberFormatter (see GPGOOGLEVISUALIZATIONAPI-10).

May 26, 2012 (version 0.5.2)

- Added isStacked attribute for ComboCoreChart.

May 17, 2012 (version 0.5.1)

- Provided support for marker style in GeoChart (see GPGOOGLEVISUALIZATIONAPI-7)
- Upgrade to Grails 2.0.1 and Release plugin 2.0.0.

January 29, 2012 (version 0.5)

- Added new visualizations: Bubble Chart, Stepped Area Chart.
- Allow all visualizations to use String data type for the attributes height and width (see GPGOOGLEVISUALIZATIONAPI-5).
- Allow all visualizations to use formatters (see GPGOOGLEVISUALIZATIONAPI-6).
- Provided taglib for Google JavaScript API import.

December 3, 2011 (version 0.4.3)

- Ready event has to be defined before calling draw on visualization (see GPGOOGLEVISUALIZATIONAPI-3).

October 2, 2011 (version 0.4.2)

- Ready event has to be defined before calling draw on visualization (see GPGOOGLEVISUALIZATIONAPI-2).

August 27, 2011 (version 0.4.1)

- Updated core chart parameters (see Google's release notes).

May 20, 2011 (version 0.4)

- Added new visualizations: Geo Chart, Candlestick Chart and Combo Chart.
- Exposed visualization parameter to set API version.

May 17, 2011 (version 0.3.1)

- Updated core chart parameters (see Google's release notes).

March 20, 2011 (version 0.3)

- Updated core chart parameters (see Google's release notes).
- Added Candlestick image chart.

November 10, 2010 (version 0.2.4)

- Updated core chart parameters (see Google's release notes).
- Improved String data type handling (see GPGOOGLEVISUALIZATIONAPI-1).

August 26, 2010 (version 0.2.3)

- Updated core chart parameters (see Google's release notes).

August 4, 2010 (version 0.2.2)

- Exposed visualization data JavaScript object (google.visualization.DataTable).
- Bugfixes: String parameters and cell labels needed to be escaped.

July 8, 2010 (version 0.2.1)

- Added onmouseover/onmouseout events to treemap.
- Support for dynamic loading.

June 5, 2010 (version 0.2)

- Added new visualization Tree Map.
- Added image charts: Pie Chart, Bar Chart, Area Chart, Line Chart and Sparkline.
- Map now provides the Terrain type.
- Support for cell object.

May 27, 2010 (version 0.1)

- Initial version.

1.4. Acknowledgments

Many thanks to all the users who reported issues and sent pull requests.

1.4.1. Authors and Contributors

- Benjamin Muschko
- Angel Ruiz
- Mansi Arora
- Robert Oschwald
- Uday Pratap Singh
- Tom Crossland
- Sdonyh
- Ryan
- Lewis Norton
- Zoran Stojakovic
- Puneet Behl
- Vmille

1.5. License

This plugin is released under the Apache License, Version 2.0

Chapter 2. Usage

2.1. Installation

2.1.1. For Grails 3.x

Add the following dependency under `build.gradle`:

```
compile "org.grails.plugins:grails-google-visualization:2.0"
```

2.1.2. For Grails 2.x

Add the following plugin under plugins in BuildConfig.groovy:

```
compile ":google-visualization:1.0.1"
```

2.2. Basic usage

- The page you want to use the visualization in has to import the Google visualization API JavaScript library. You can do so by using the taglib `<gvisualization:apiImport/>` or by importing it using the HTML script tag.

```
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
```

- All visualizations in the taglib use the namespace **gvisualization**.
- Apart from the configuration options specific to a visualization (see visualization specifications) there are multiple attributes that you have to set for your visualization.
- ***name*** (optional) - JavaScript variable name for visualization object (defaults to "visualization").
- ***version*** (optional) - API version for visualization object (defaults to "1").
- ***elementId*** (required) - HTML div ID used to render visualization.
- ***dynamicLoading*** (optional) - Renders visualization over dynamic loading - required when rendered in AJAX response (defaults to false).
- ***language*** (optional) - Forces localized version of visualization. The language property is a two-letter ISO 639-1 language code.
- ***columns*** (required) - List of column data types and names to be displayed in the visualization.
- ***data*** (required) - List of data to be displayed for columns.
  - To express objects you use the Map notation inside an EL expression. Eg: [source,groovy] ---- legend="${[position: 'top', alignment: 'center']}" ----
  - If your data requires the usage of the cell object you can import and populate the class

`org.grails.plugins.google.visualization.data.Cell`.

2.3. Events

If you want to register an event handler for your visualization you can by adding an event attribute. The value you give that attribute is the name of the JavaScript function acting as callback handler. Please check the visualization specification for available event names.

The variable name of the visualization JavaScript object by default is "**visualization**", the name of the `google.visualization.DataTable` object is "**visualization_data**". You can always change the names by setting the taglib attribute **"name"**.

Example:

```
<%
def employeeColumns = [['string', 'Name'], ['string', 'Salary'], ['boolean', 'Full
Time Employee']]
def employeeData = [['Mike', '$10,000', true], ['Jim', '$8,000', false], ['Alice',
'$12,500', true],
['Bob', '$7,000', true]]
%>

<script type="text/javascript">
  function selectHandler(e) {
    alert('A table row was selected');
  }
</script>

<gvisualization:table elementId="table" width="${400}" height="${130}"
columns="${employeeColumns}"
data="${employeeData}" select="selectHandler" />
<div id="table"></div>
```

Since the **Table visualization** has both a "page" config option and a "page" event, to use the latest the taglib attribute is called **"page-event"**.

2.4. Formatters

You can apply formatters to all visualizations using the "formatters" taglib attribute as the underlying implementation is a google.visualization.DataTable.

However, using formatters makes the most sense for the Table visualization.

A full set of examples can be found on this GSP page.The value you have to pass in is a list of classes implementing the `org.grails.plugins.google.visualization.formatter.Formatter` interface. The implementations you can apply are the following:

```
org.grails.plugins.google.visualization.formatter.PatternFormatter
org.grails.plugins.google.visualization.formatter.NumberFormatter
org.grails.plugins.google.visualization.formatter.DateFormatter
org.grails.plugins.google.visualization.formatter.ColorRange
org.grails.plugins.google.visualization.formatter.ColorFormatter
org.grails.plugins.google.visualization.formatter.BarFormatter
org.grails.plugins.google.visualization.formatter.ArrowFormatter
```

Example:

```
<%@ page import="org.grails.plugins.google.visualization.formatter.BarFormatter" %>
<%
def departmentRevenueColumns = [['string', 'Department'], ['number', 'Revenues']]
def departmentRevenueData = [['Shoes', 10700], ['Sports', -15400], ['Toys', 12500],
['Electronics', -2100],
['Food', 22600], ['Art', 1100]]
def barFormatter = new BarFormatter(1)
barFormatter.width = 120
def barFormatters = [barFormatter]
%>

<gvisualization:table elementId="barformat_div" allowHtml="${true}"
showRowNumber="${true}"
columns="${departmentRevenueColumns}" data="${departmentRevenueData}"
formatters="${barFormatters}"/>
<div id="barformat_div"></div>
```

Chapter 3. Charts

3.1. Core Chart

3.1.1. Pie Chart

VisualizationController.groovy

```
class VisualizationController {
  def pieChart() {
   List myDailyActivitiesColumns = [['string', 'Task'], ['number', 'Hours per Day']]
   List myDailyActivitiesData = [['Work', 11], ['Eat', 2], ['Commute', 2], ['Watch
TV', 2], ['Sleep', 7]]
   render template: "pieChart", model: ["myDailyActivitiesColumns":
myDailyActivitiesColumns,
    "myDailyActivitiesData": myDailyActivitiesData]
  }
}
```

_pieChart.gsp

```
<gvisualization:pieCoreChart elementId="piechart" title="My Daily Activities"
width="${450}" height="${300}"
columns="${myDailyActivitiesColumns}" data="${myDailyActivitiesData}" />
<div id="piechart"></div>
```

3.1.2. Bar Chart

VisualizationController.groovy

```
class VisualizationController {
  def barChart() {
    List companyPerformanceColumns = [['string', 'Year'], ['number', 'Sales'],
['number', 'Expenses']]
    List companyPerformanceData = [['2004', 1000, 400], ['2005', 1170, 460], ['2006',
660, 1120], ['2007', 1030, 540]]
    render template: "barChart", model: ["companyPerformanceColumns":
companyPerformanceColumns, "companyPerformanceData": companyPerformanceData]
  }
}
```

_barChart.gsp

```
 <gvisualization:barCoreChart elementId="barchart" title="Company Performance"
width="${400}" height="${240}" vAxis="${[title: 'Year', titleColor: 'red']}"
columns="${companyPerformanceColumns}" data="${companyPerformanceData}" />
 <div id="barchart"></div>
```

3.1.3. Bubble Chart

VisualizationController.groovy

```
class VisualizationController {
  def bubbleChart() {
    List lifeExpectancyFertilityRateData = [['CAN', 80.66, 1.67, 'North America',
33739900], ['DEU', 79.84, 1.36, 'Europe', 81902307], ['DNK', 78.6, 1.84, 'Europe',
5523095], ['EGY', 72.73, 2.78, 'Middle East', 79716203], ['GBR', 80.05, 2, 'Europe',
61801570], ['IRN', 72.49, 1.7, 'Middle East', 73137148], ['IRQ', 68.09, 4.77, 'Middle
East', 31090763], ['ISR', 81.55, 2.96, 'Middle East', 7485600], ['RUS', 68.6, 1.54,
'Europe', 141850000], ['USA', 78.09, 2.05, 'North America', 307007000]]
    render template: "bubbleChart", model: ["lifeExpectancyFertilityRateData":
lifeExpectancyFertilityRateData]
  }
}
```

_bubbleChart.gsp

```
<gvisualization:bubbleCoreChart elementId="bubblechart" title="Correlation between
life expectancy, fertility rate and population of some world countries (2010)"
hAxis="${[title: 'Life Expectancy']}" vAxis="${[title: 'Fertility Rate']}"
bubble="${[textStyle: '{fontSize: 11}']}"
columns="${lifeExpectancyFertilityRateColumns}"
data="${lifeExpectancyFertilityRateData}" />
```

3.1.4. Column Chart

3.1.5. Area Chart

3.1.6. Line Chart

3.1.7. Scatter Chart

3.1.8. Stepped Area Chart

3.1.9. Candlestick Chart

3.1.10. Combo Chart

3.2. Additional Charts

3.2.1. Gauge

3.2.2. Table

3.2.3. Map

3.2.4. Annotated Time Line

3.2.5. Annotation Chart

3.2.6. Org Chart

3.2.7. Intensity Map

3.2.8. Geo Map

3.2.9. Geo Chart

3.2.10. Motion Chart

3.2.11. Tree Map

3.2.12. Timeline

3.2.13. Calendar Chart

Unresolved directive in ref/charts/additional.adoc - include::additional/core/dataTableRole.adoc[]

Chapter 4. Visualizations

4.1. Available Visualizations

Working examples for all available visualization can be found on this GSP page to this page on GitHub.

4.1.1. Interactive Charts

- Pie Core Chart: `<gvisualization:pieCoreChart/>`
- Bar Core Chart: `<gvisualization:barCoreChart/>`
- Bubble Core Chart: `<gvisualization:bubbleCoreChart/>`
- Column Core Chart: `<gvisualization:columnCoreChart/>`
- Area Core Chart: `<gvisualization:areaCoreChart/>`
- Line Core Chart: `<gvisualization:lineCoreChart/>`
- Scatter Core Chart: `<gvisualization:scatterCoreChart/>`
- Stepped Area Core Chart: `<gvisualization:steppedAreaCoreChart/>`
- Candlestick Core Chart: `<gvisualization:candlestickCoreChart/>`
- Combo Core Chart: `<gvisualization:comboCoreChart/>`
- Gauge: `<gvisualization:gauge/>`
- Table: `<gvisualization:table/>`
- Map: `<gvisualization:map/>`
- Annotated Time Line: `<gvisualization:annotatedTimeLine/>`
- Organizational Chart: `<gvisualization:orgChart/>`
- Intensity Map: `<gvisualization:intensityMap/>`
- Geo Map: `<gvisualization:geoMap/>`
- Geo Chart: `<gvisualization:geoChart/>`
- Motion Chart: `<gvisualization:motionChart/>`
- Tree Map: `<gvisualization:treeMap/>`
- Timeline: `<gvisualization:timeLine/>`
- Calendar Chart: `<gvisualization:calendarChart/>`

4.1.2. Deprecated Charts

- Pie Chart: `<gvisualization:pieChart/>`
- Bar Chart: `<gvisualization:barChart/>`
- Column Chart: `<gvisualization:columnChart/>`
- Area Chart: `<gvisualization:areaChart/>`

- Line Chart: `<gvisualization:lineChart/>`
- Scatter Chart: `<gvisualization:scatterChart/>`