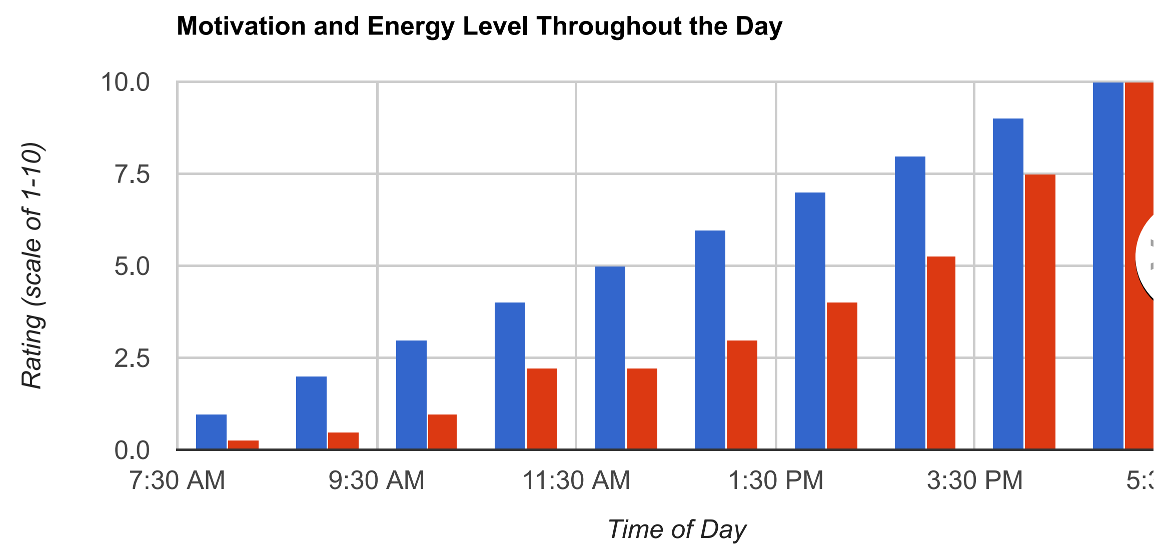
Visualization: Column Chart

Overview

A *column chart* is a vertical bar chart rendered in the browser using SVG (http://www.w3.org/Graphics/SVG/) or VML (http://en.wikipedia.org/wiki/Vector_Markup_Language), whichever is appropriate for the user's browser. Like all Google charts, column charts display tooltips when the user hovers over the data. For a horizontal version of this chart, see the bar chart (https://developers.google.com/chart/interactive/docs/gallery/barchart).

Examples

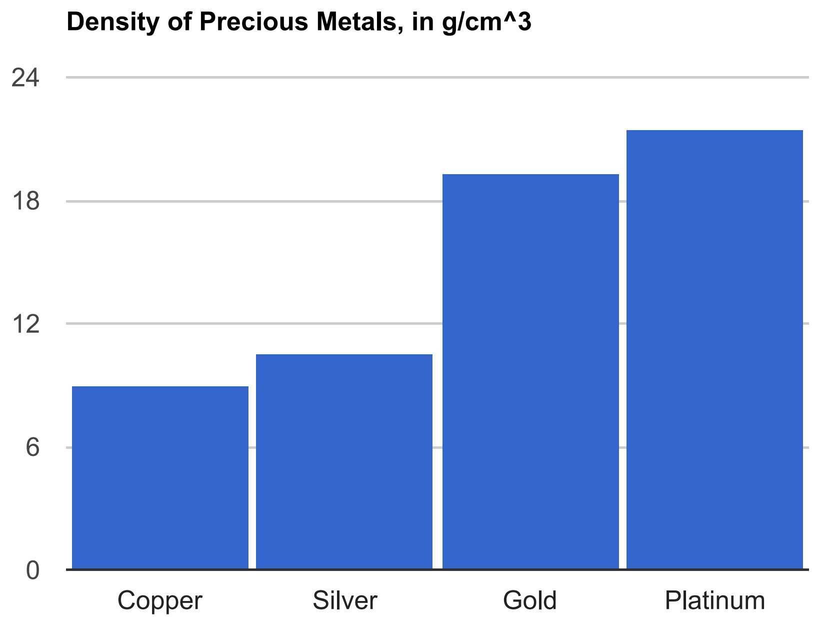
Basic column chart with multiple series



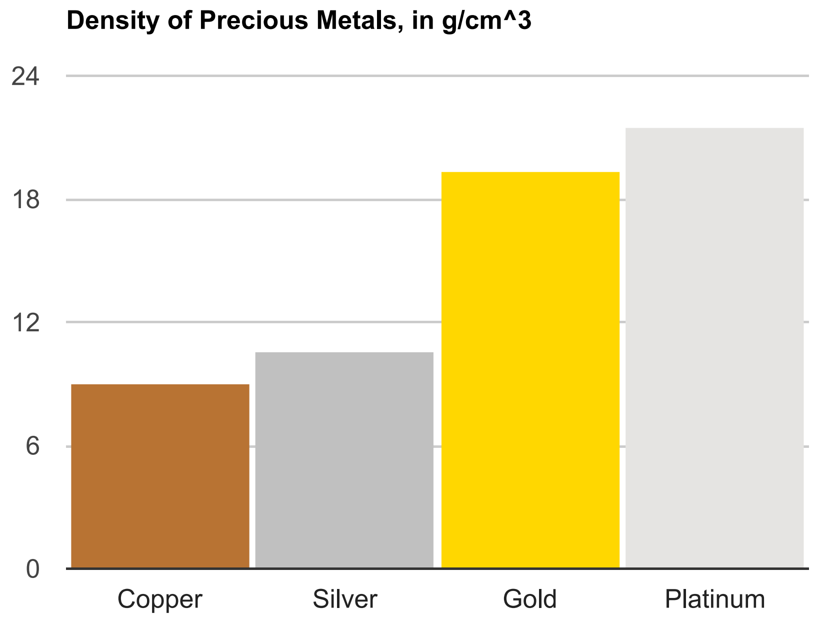
CODE IT YOURSELF ON JSFIDDLE

Coloring columns

Let's chart the densities of four precious metals:



Above, all colors are the default blue. That's because they're all part of the same series; if there were a second series, that would have been colored red. We can customize these colors with the *style role* (https://developers.google.com/chart/interactive/docs/roles#stylerole):



There are three different ways to choose the colors, and our data table showcases them all: RGB values, English color names, and a CSS-like declaration:

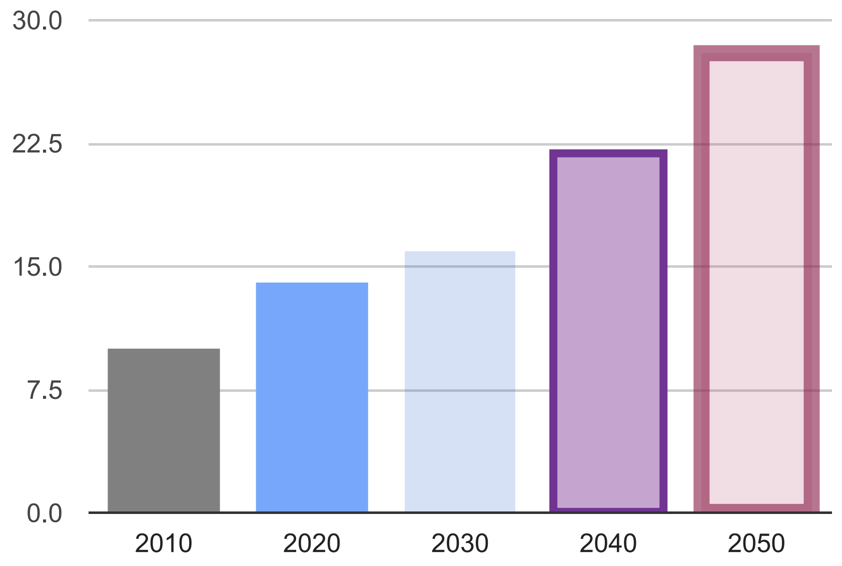
```
       var data = google.visualization.arrayToDataTable([
         ['Element', 'Density', { role: 'style' }],
         ['Copper', 8.94, '#b87333'],            // RGB value
         ['Silver', 10.49, 'silver'],            // English color name
         ['Gold', 19.30, 'gold'],

       ['Platinum', 21.45, 'color: #e5e4e2' ], // CSS-style declaration
      ]);
```

Column styles

The style role (https://developers.google.com/chart/interactive/docs/roles#stylerole) lets your control several aspects of column appearance with CSS-like declarations:

- `color`
- `opacity`
- `fill-color`
- `fill-opacity`
- `stroke-color`
- `stroke-opacity`
- `stroke-width`

We don't recommend that you mix styles too freely inside a chart—pick a style and stick with it—but to demonstrate all the style attributes, here's a sampler:



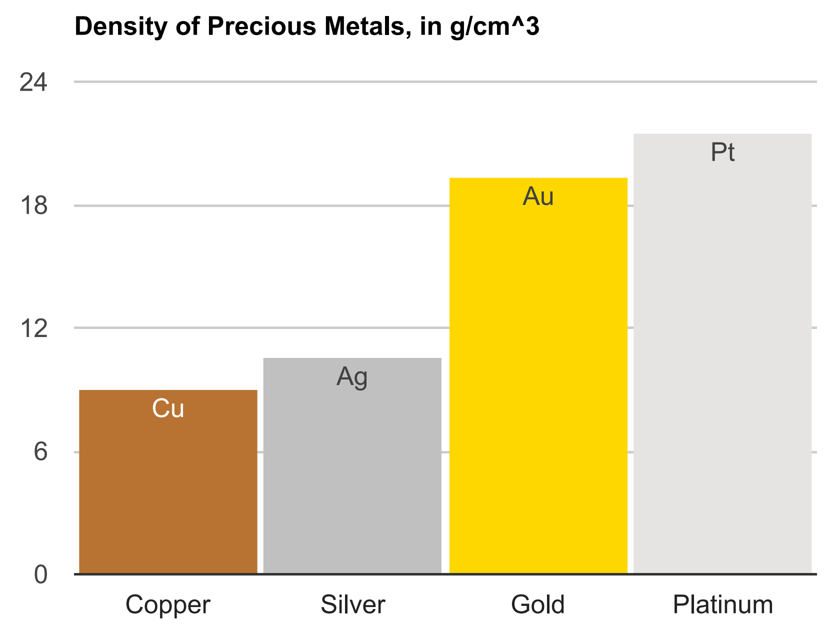
The first two columns each use a specific `color` (the first with an English name, the second with an RGB value). No `opacity` was chosen, so the default of 1.0 (fully opaque) is used; that's why the second column obscures the gridline behind it. In the third column, an `opacity` of 0.2 is used, revealing the gridline. In the fourth, three style attributes are used: `stroke-color` and `stroke-width` to draw the border, and `fill-color` to specify the color of the rectangle inside. The rightmost column additionally uses `stroke-opacity` and `fill-opacity` to choose opacities for the border and fill:

```
function drawChart() {
   var data = google.visualization.arrayToDataTable([
     ['Year', 'Visitations', { role: 'style' } ],
     ['2010', 10, 'color: gray'],
     ['2020', 14, 'color: #76A7FA'],
     ['2030', 16, 'opacity: 0.2'],
     ['2040', 22, 'stroke-color: #703593; stroke-width: 4; fill-color: #C5
     ['2050', 28, 'stroke-color: #871B47; stroke-opacity: 0.6; stroke-widt
   ]);
```

Labeling columns

Charts have several kinds of labels, such as tick labels, legend labels, and labels in the tooltips. In this section, we'll see how to put labels inside (or near) the columns in a column chart.

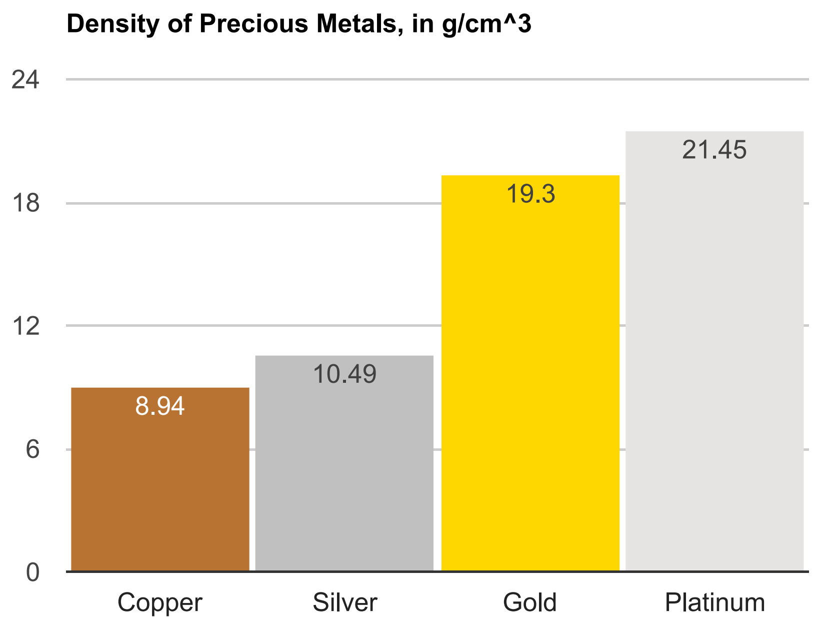
Let's say we wanted to annotate each column with the appropriate chemical symbol. We can do that with the *annotation* role:



In our data table, we define a new column with `{ role: 'annotation' }` to hold our column labels:

```
      var data = google.visualization.arrayToDataTable([
        ['Element', 'Density', { role: 'style' }, { role: 'annotation' } ],
        ['Copper', 8.94, '#b87333', 'Cu' ],
        ['Silver', 10.49, 'silver', 'Ag' ],
        ['Gold', 19.30, 'gold', 'Au' ],
        ['Platinum', 21.45, 'color: #e5e4e2', 'Pt' ]
      ]);
```

While users can hover over the columns to see the data values, you might want to include them on the columns themselves:



This is a little more complicated than it should be, because we create a `DataView` to specify the annotation for each column.

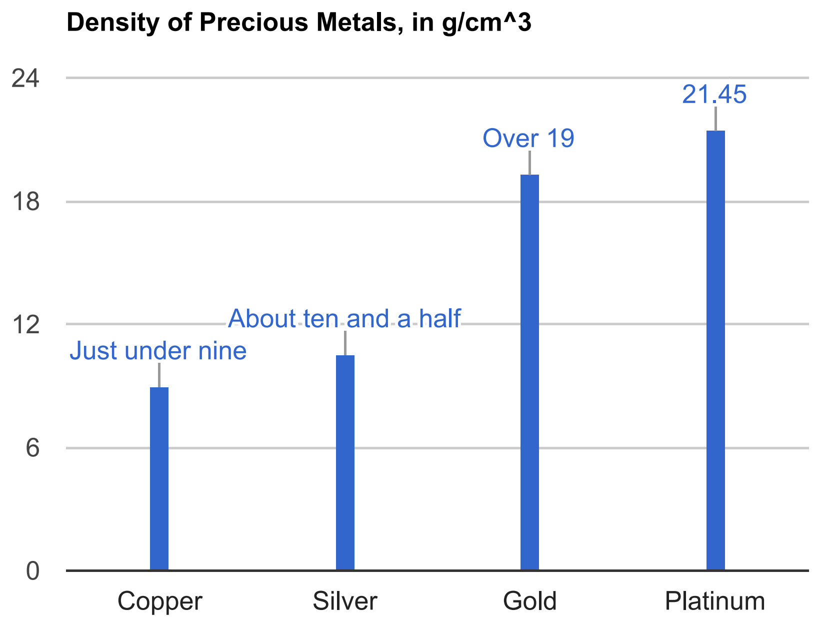
```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.j
<script type="text/javascript">
  google.charts.load("current", {packages:['corechart']});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var data = google.visualization.arrayToDataTable([
      ["Element", "Density", { role: "style" } ],
      ["Copper", 8.94, "#b87333"],
      ["Silver", 10.49, "silver"],
      ["Gold", 19.30, "gold"],
      ["Platinum", 21.45, "color: #e5e4e2"]
    ]);

    var view = new google.visualization.DataView(data);
    view.setColumns([0, 1,
                     { calc: "stringify",
                       sourceColumn: 1,
                       type: "string",
                       role: "annotation" },
                     2]);

    var options = {
      title: "Density of Precious Metals, in g/cm^3",
      width: 600,
        height: 400,
        bar: {groupWidth: "95%"},
        legend: { position: "none" },
      };
      var chart = new google.visualization.ColumnChart(document.getElementByI
      chart.draw(view, options);
  }
  </script>
<div id="columnchart_values" style="width: 900px; height: 300px;"></div>
```

If we wanted to format the value differently, we could define a formatter (//developers.google.com/chart/interactive/docs/reference#formatters) and wrap it in a function like this:

```
      function getValueAt(column, dataTable, row) {
        return dataTable.getFormattedValue(row, column);
      }
```

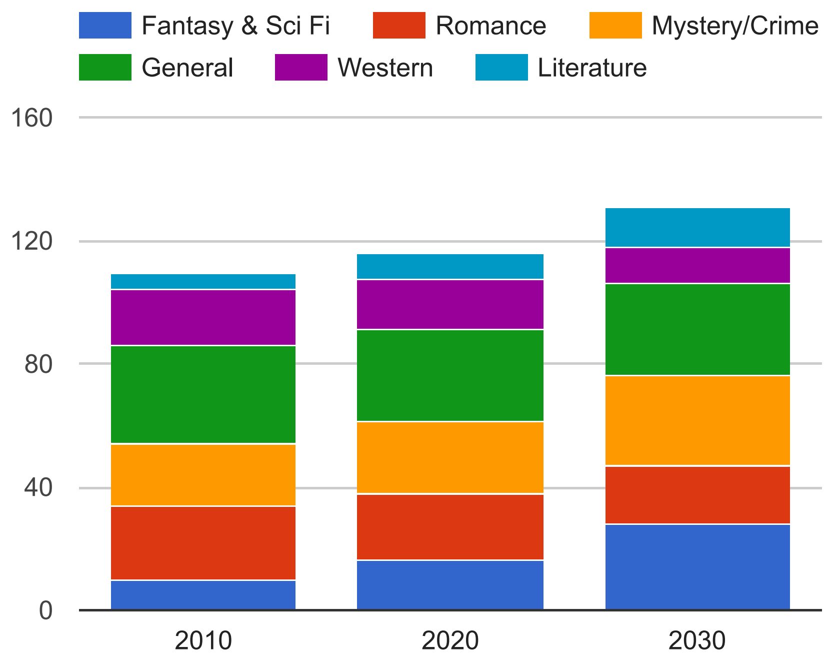
Then we could call it with `calc: getValueAt.bind(undefined, 1)`.

If the label is too big to fit entirely inside the column, it's displayed outside:



Stacked column charts

A *stacked column chart* is a column chart that places related values atop one another. If there are any negative values, they are stacked in reverse order below the chart's baseline. It's typically used when a category naturally divides into components. For instance, consider some hypothetical book sales, divided by genre and compared across time:

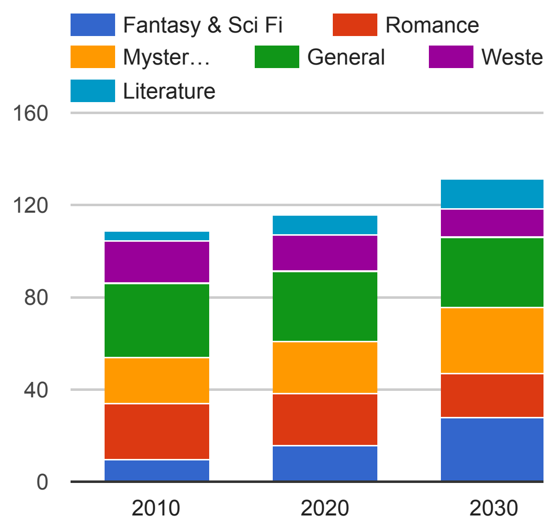


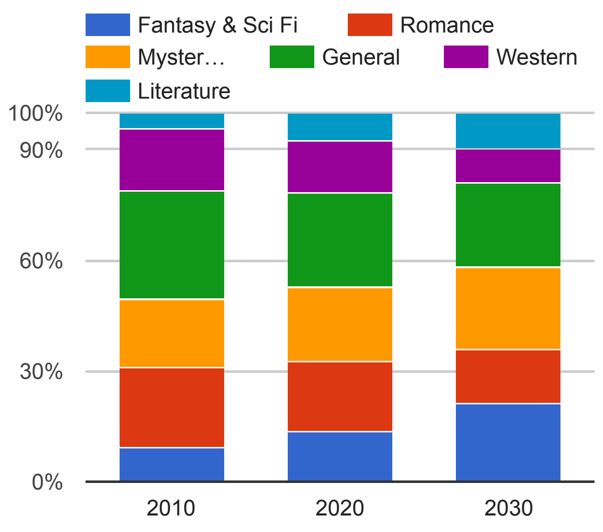
You create a stacked column chart by setting the `isStacked` option to `true`:

```
      var data = google.visualization.arrayToDataTable([
        ['Genre', 'Fantasy & Sci Fi', 'Romance', 'Mystery/Crime', 'General',
         'Western', 'Literature', { role: 'annotation' } ],
        ['2010', 10, 24, 20, 32, 18, 5, ''],
        ['2020', 16, 22, 23, 30, 16, 9, ''],
        ['2030', 28, 19, 29, 30, 12, 13, '']
      ]);

      var options = {
        width: 600,
        height: 400,
        legend: { position: 'top', maxLines: 3 },
        bar: { groupWidth: '75%' },
        isStacked: true,
      };
```

Stacked column charts also support 100% stacking, where the stacks of elements at each domain-value are rescaled such that they add up to 100%. The options for this are `isStacked: 'percent'`, which formats each value as a percentage of 100%, and `isStacked: 'relative'`, which formats each value as a fraction of 1. There is also an `isStacked: 'absolute'` option, which is functionally equivalent to `isStacked: true`.

Note in the 100% stacked chart on the right, the tick values are based on the relative 0-1 scale as fractions of 1, but the axis values are displayed as percentages. This is because the percentage axis ticks are the result of applying a format of "#.##%" to the relative 0-1 scale values. When using `isStacked: 'percent'`, be sure to specify any ticks/axis values using the relative 0-1 scale.





STACKED   100% STACKED

```
var options_fullStacked = {
  isStacked: 'percent',
  height: 300,
  legend: {position: 'top', maxLines: 3},
  vAxis: {
    minValue: 0,
    ticks: [0, .3, .6, .9, 1]
  }
};
```

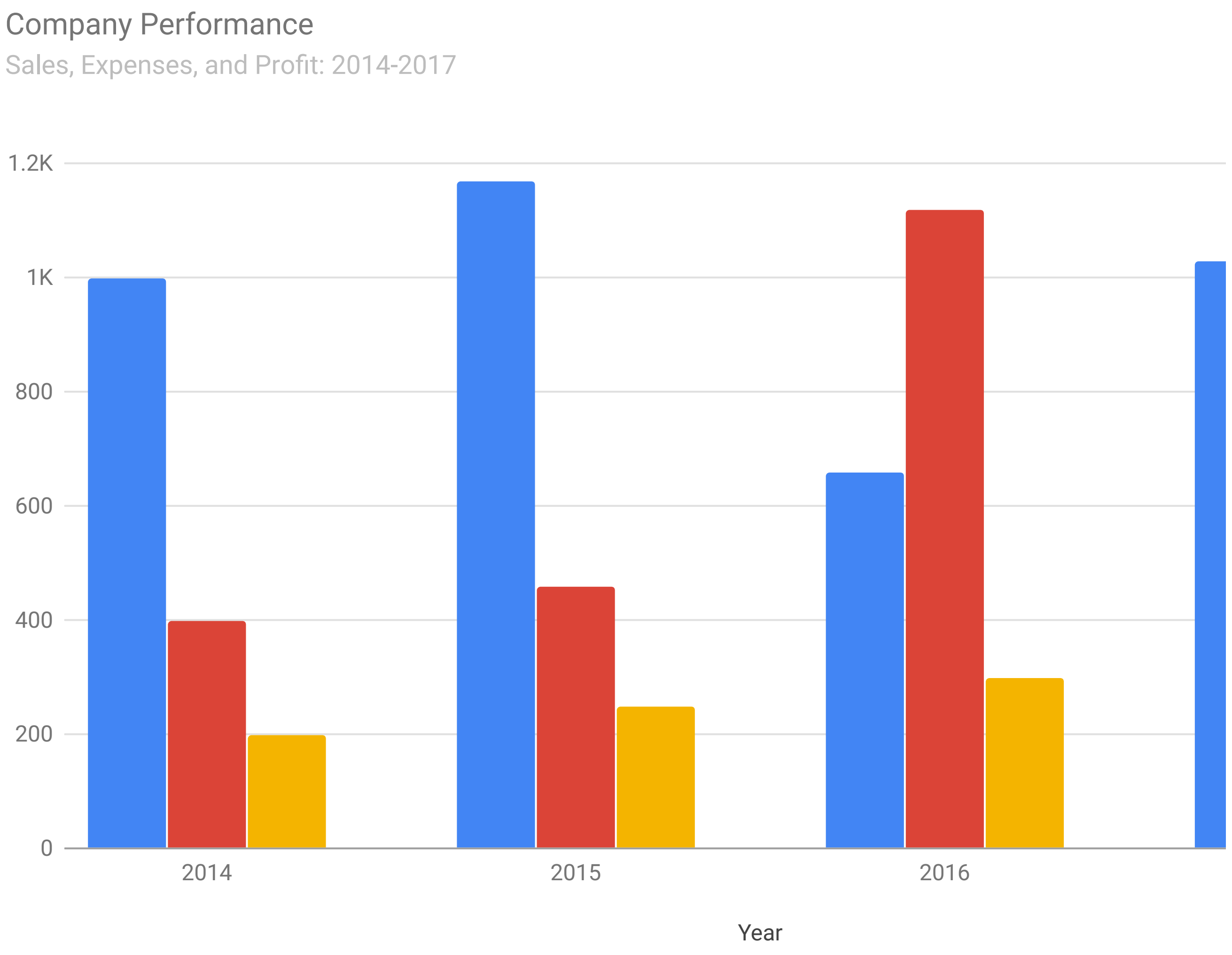
Creating Material column charts

In 2014, Google announced guidelines intended to support a common look and feel across its properties and apps (such as Android apps) that run on Google platforms. We call this effort *Material Design*. We'll be providing "Material" versions of all our core charts; you're welcome to use them if you like how they look.

Creating a Material Column Chart is similar to creating what we'll now call a "Classic" Column Chart. You load the Google Visualization API (although with the `'bar'` package instead of the `'corechart'` package), define your datatable, and then create an object (but of class `google.charts.Bar` instead of `google.visualization.ColumnChart`).

Since bar charts and column charts are essentially identical but for orientation, we call both Material Bar Charts, regardless of whether the bars are vertical (classically, a column chart) or horizontal (a bar chart). In Material, the only difference is in the `bars` option. When set to `'horizontal'`, the orientation will resemble the traditional Classic Bar Chart; otherwise, the bars will be vertical.

Note: Material Charts will not work in old versions of Internet Explorer. (IE8 and earlier versions don't support SVG, which Material Charts require.)



Material Column Charts have many small improvements over Classic Column Charts, including an improved color palette, rounded corners, clearer label formatting, tighter default spacing between series, softer gridlines and titles (and the addition of subtitles).

```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader
    <script type="text/javascript">
      google.charts.load('current', {'packages':['bar']});
      google.charts.setOnLoadCallback(drawChart);

      function drawChart() {
        var data = google.visualization.arrayToDataTable([
          ['Year', 'Sales', 'Expenses', 'Profit'],
          ['2014', 1000, 400, 200],
          ['2015', 1170, 460, 250],
          ['2016', 660, 1120, 300],
          ['2017', 1030, 540, 350]
        ]);

        var options = {
          chart: {
            title: 'Company Performance',
            subtitle: 'Sales, Expenses, and Profit: 2014-2017',
          }
        };

        var chart = new google.charts.Bar(document.getElementById('columnchar

        chart.draw(data, options);
      }
    </script>
  </head>
  <body>
    <div id="columnchart_material" style="width: 900px; height: 500px;"></div
  </body>
</html>
```

The Material Charts are in **beta**. The appearance and interactivity are largely final, but many of the options available in Classic Charts are not yet available in them. You can find a list of options that are not yet supported in this issue (https://github.com/google/google-visualization-issues/issues/2143).

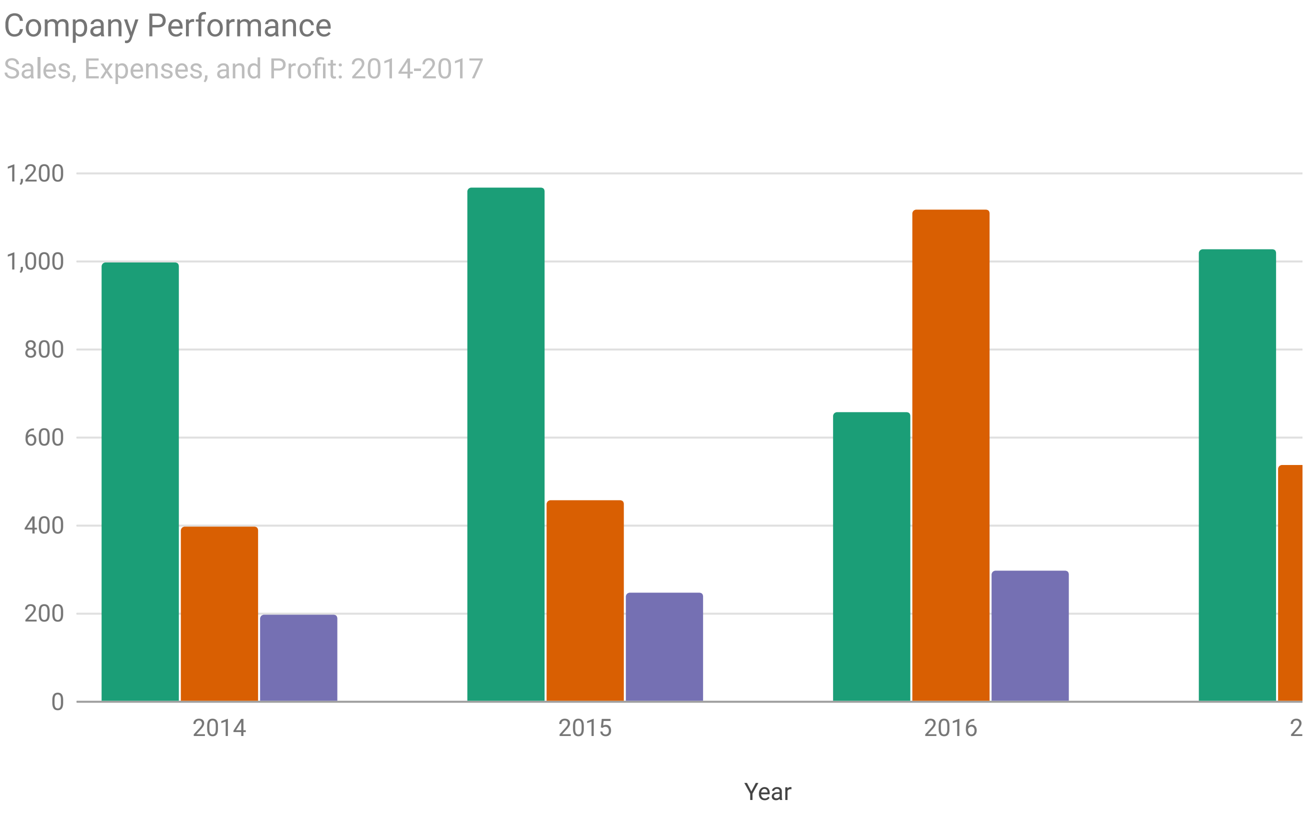
Also, the way options are declared is not finalized, so you must convert your options by replacing this line:

```
chart.draw(data, options);
```

...with this:

```
chart.draw(data, google.charts.Bar.convertOptions(options));
```

Using `google.charts.Bar.convertOptions()` allows you to take advantage of certain features, such as the `hAxis/vAxis.format` preset options.

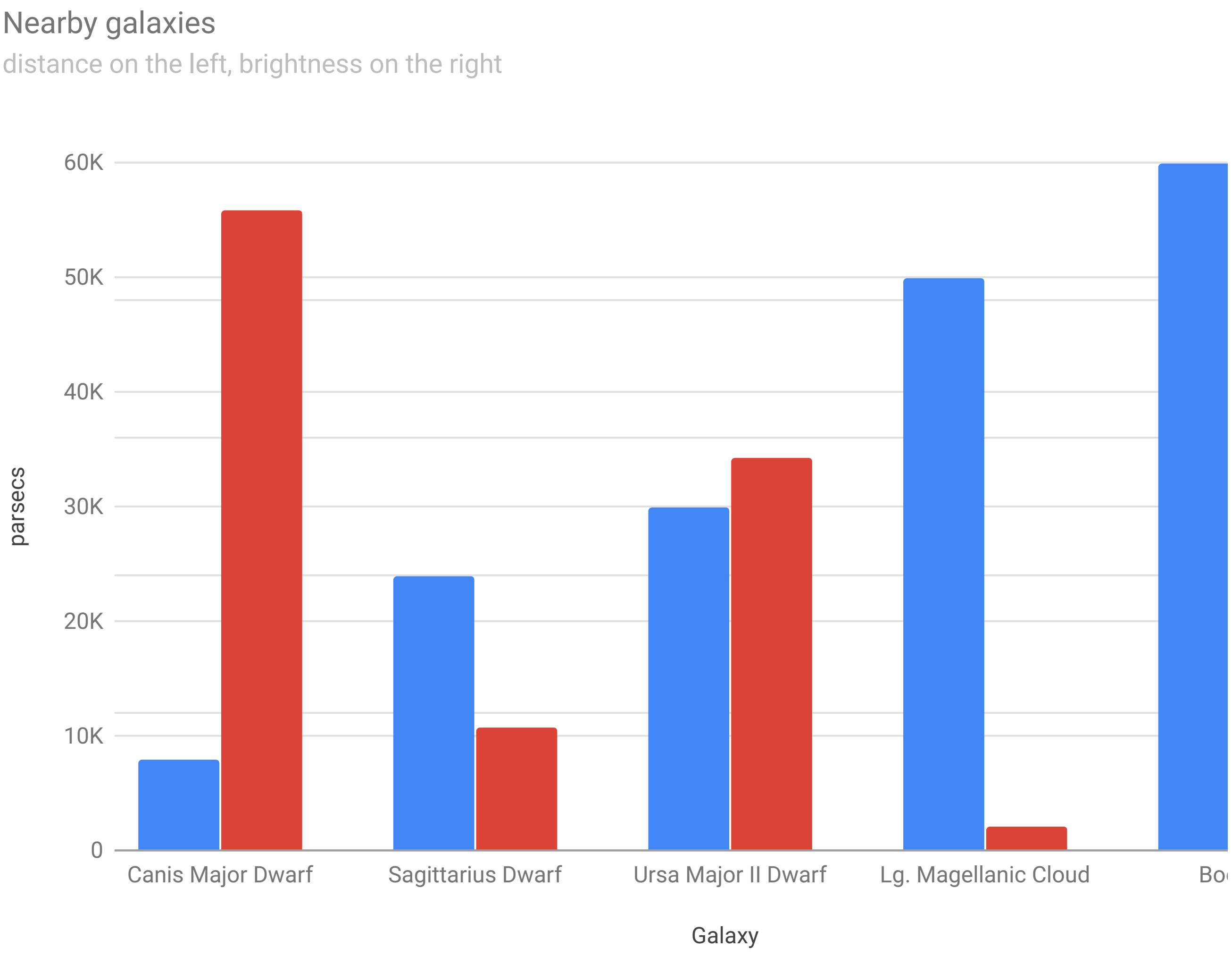


CODE IT YOURSELF ON JSFIDDLE

Dual-Y charts

Note: Dual-Y axes are available only for Material charts (i.e., those with package `bar`).

Sometimes you'll want to display two series in a column chart, with two independent Y-axes: a left axis for one series, and a right axis for another:



CODE IT YOURSELF ON JSFIDDLE

Note that not only are our two y-axes labeled differently ("parsecs" versus "apparent magnitude") but they each have their own independent scales and gridlines. If you want to customize this behavior, use the `vAxis.gridlines` options.

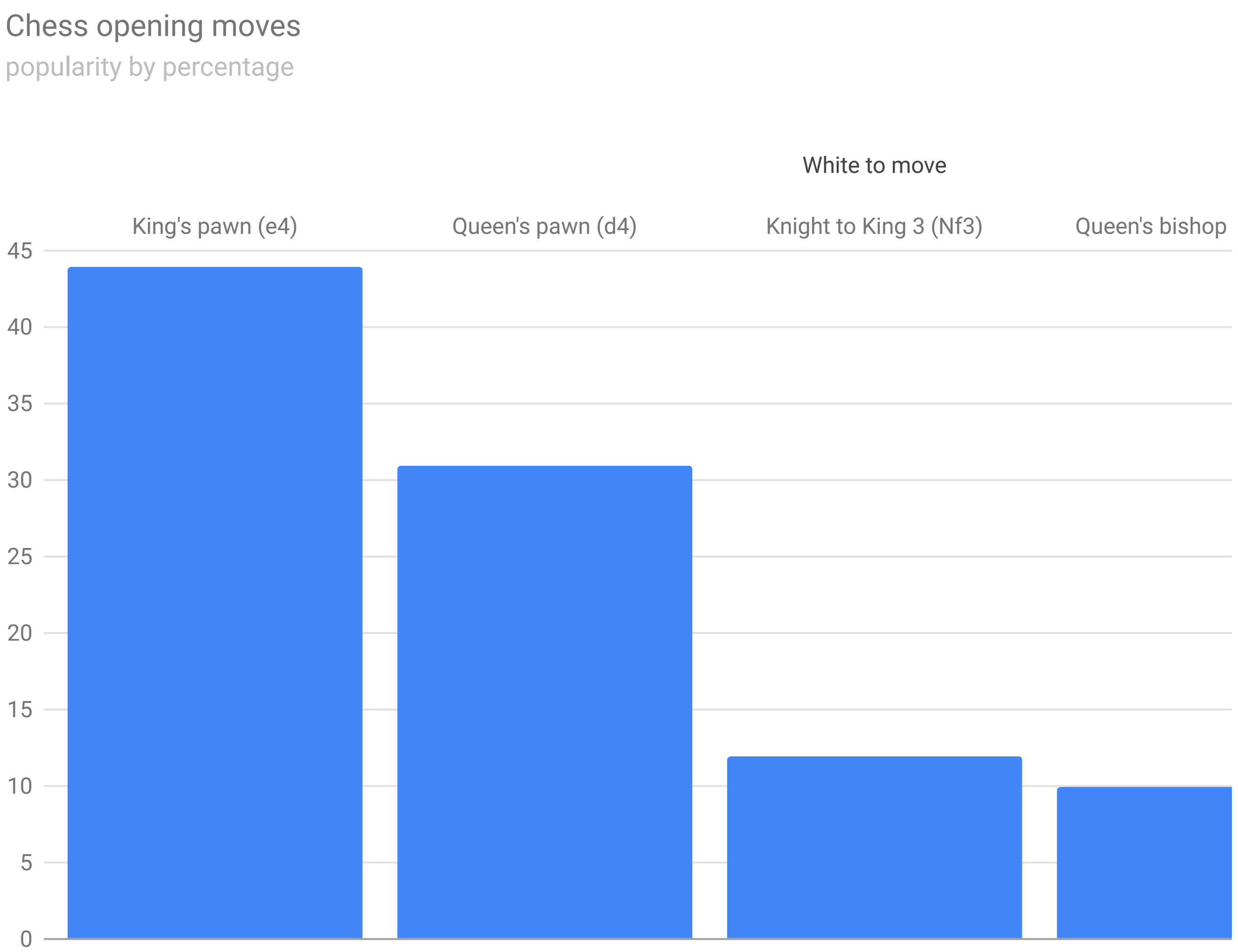
In the code below, the `axes` and `series` options together specify the dual-Y appearance of the chart. The `series` option specifies which axis to use for each (`'distance'` and `'brightness'`; they needn't have any relation to the column names in the datatable). The `axes` option then makes this chart a dual-Y chart, placing the `'distance'` axis on the left (labeled "parsecs") and the `'brightness'` axis on the right (labeled "apparent magnitude").

```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader
    <script type="text/javascript">
      google.charts.load('current', {'packages':['bar']});
      google.charts.setOnLoadCallback(drawStuff);
      function drawStuff() {
        var data = new google.visualization.arrayToDataTable([
          ['Galaxy', 'Distance', 'Brightness'],
          ['Canis Major Dwarf', 8000, 23.3],
          ['Sagittarius Dwarf', 24000, 4.5],
          ['Ursa Major II Dwarf', 30000, 14.3],
          ['Lg. Magellanic Cloud', 50000, 0.9],
          ['Bootes I', 60000, 13.1]
        ]);

        var options = {
          width: 900,
          chart: {
            title: 'Nearby galaxies',
            subtitle: 'distance on the left, brightness on the right'
          },
          series: {
            0: { axis: 'distance' }, // Bind series 0 to an axis named 'dista
            1: { axis: 'brightness' } // Bind series 1 to an axis named 'brig
          },
          axes: {
            y: {
              distance: {label: 'parsecs'}, // Left y-axis.
              brightness: {side: 'right', label: 'apparent magnitude'} // Rig
            }
          }
        };

      var chart = new google.charts.Bar(document.getElementById('dual_y_div')
      chart.draw(data, options);
    };
    </script>
  </head>
  <body>
    <div id="dual_y_div" style="width: 900px; height: 500px;"></div>
  </body>
</html>
```

Top-X charts

Note: Top-X axes are available only for Material charts (i.e., those with package `bar`).

If you want to put the X-axis labels and title on the top of your chart rather than the bottom, you can do that in Material charts with the `axes.x` option:



CODE IT YOURSELF ON JSFIDDLE

```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader
    <script type="text/javascript">
      google.charts.load('current', {'packages':['bar']});
      google.charts.setOnLoadCallback(drawStuff);

      function drawStuff() {
        var data = new google.visualization.arrayToDataTable([
          ['Move', 'Percentage'],
          ["King's pawn (e4)", 44],
          ["Queen's pawn (d4)", 31],
          ["Knight to King 3 (Nf3)", 12],
          ["Queen's bishop pawn (c4)", 10],
          ['Other', 3]
        ]);
        var options = {
          title: 'Chess opening moves',
          width: 900,
          legend: { position: 'none' },
          chart: { subtitle: 'popularity by percentage' },
          axes: {
            x: {
              0: { side: 'top', label: 'White to move'} // Top x-axis.
            }
          },
          bar: { groupWidth: "90%" }
        };

        var chart = new google.charts.Bar(document.getElementById('top_x_div'
        // Convert the Classic options to Material options.
        chart.draw(data, google.charts.Bar.convertOptions(options));
      };
    </script>
  </head>
  <body>
    <div id="top_x_div" style="width: 900px; height: 500px;"></div>
  </body>
</html>
```

Loading

The `google.charts.load` package name is `"corechart"`.

```
google.charts.load("current", {packages: ["corechart"]});
```

For Material Column Charts, the `google.charts.load` package name is `"bar"`. (Not a typo: the Material Bar Chart handles both orientations.)

```
google.charts.load("current", {packages: ["bar"]});
```

The visualization's class name is `google.visualization.ColumnChart`.

```
var visualization = new google.visualization.ColumnChart(container);
```

For Material Column Charts, the visualization's class name is `google.charts.Bar`. (Not a typo: the Material Bar Chart handles both orientations.)

```
var chart = new google.charts.Bar(container);
```

Data format

Each row in the table represents a group of adjacent bars.

Rows: Each row in the table represents a group of bars.

Columns:

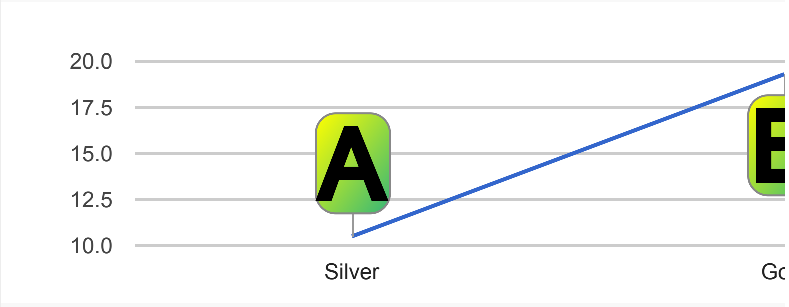
	Column 0
Purpose:	• X-axis group labels (discrete (https://developers.google.com/chart/i) • X-axis values (continuous (https://developers.google.com/chart/i)
Data Type:	• string (discrete (https://developers.google.com/chart/i) • number, date, datetime or timeofday (continuous (https://developers.google.com/chart/i)
Role:	domain
Optional column roles (https://developers.google.com/chart/interactive/docs/roles) :	*None*

Configuration options

Name	
animation.duration	The duration of the animation, in milliseconds. For details, see t (https://developers.google.com/chart/interactive/docs/animat **Type:** number **Default:** 0
animation.easing	The easing function applied to the animation. The following opt • 'linear' - Constant speed. • 'in' - Ease in - Start slow and speed up. • 'out' - Ease out - Start fast and slow down. • 'inAndOut' - Ease in and out - Start slow, speed up, then slow **Type:** string **Default:** 'linear'
animation.startup	Determines if the chart will animate on the initial draw. If **`true`**, animate to its final state. **Type:** boolean **Default** false
annotations.alwaysOutside	In Bar (https://developers.google.com/chart/interactive/docs/g (https://developers.google.com/chart/interactive/docs/gallery/ draws all annotations outside of the Bar/Column. **Type:** boolean **Default:** false

annotations.boxStyle	For charts that support annotations (https://developers.google. **`annotations.boxStyle`** object controls the appearance of th
	
	`var options = {`
	`  annotations: {`
	`    boxStyle: {`
	`      // Color of the box outline.`
	`      stroke: '#888',`
	`      // Thickness of the box outline.`
	`      strokeWidth: 1,`
	`      // x-radius of the corner curvature.`
	`      rx: 10,`
	`      // y-radius of the corner curvature.`
	`      ry: 10,`
	`      // Attributes for linear gradient fill.`
	`      gradient: {`
	`        // Start color for gradient.`
	`        color1: '#fbf6a7',`

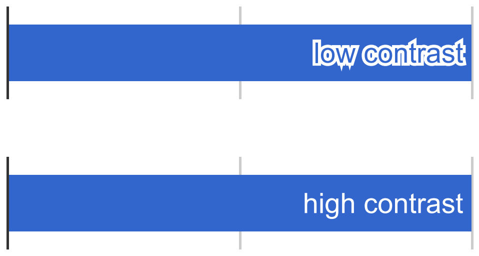
```
        // Finish color for gradient.
        color2: '#33b679',
        // Where on the boundary to start and
        // end the color1/color2 gradient,
        // relative to the upper left corner
        // of the boundary.
        x1: '0%', y1: '0%',
        x2: '100%', y2: '100%',
        // If true, the boundary for x1,
        // y1, x2, and y2 is the box. If
        // false, it's the entire chart.
        useObjectBoundingBoxUnits: true
      }
    }
  }
};
```



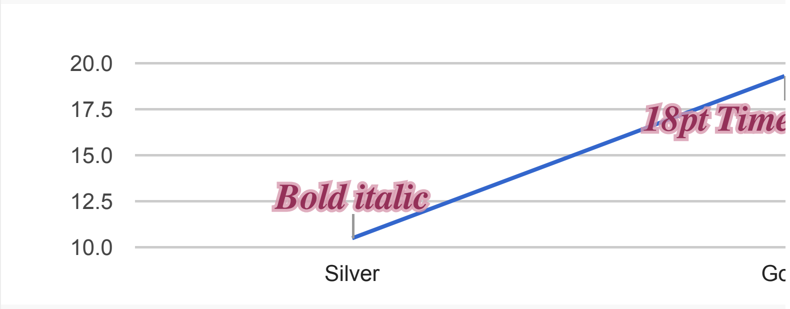
This option is currently supported for area, bar, column, combo,
supported by the Annotation Chart
(https://developers.google.com/chart/interactive/docs/gallery,

Type: object
Default: null

annotations.datum	For charts that support annotations (https://developers.google. **`annotations.datum`** object lets you override Google Charts' c individual data elements (such as values displayed with each ba color with **`annotations.datum.stem.color`**, the stem lengt **`annotations.datum.stem.length`**, and the style with **`annc`** **Type:** object **Default:** color is "black"; length is 12; style is "point".
annotations.domain	For charts that support annotations (https://developers.google. **`annotations.domain`** object lets you override Google Charts' domain (the major axis of the chart, such as the X axis on a typi with **`annotations.domain.stem.color`**, the stem length wi **`annotations.domain.stem.length`**, and the style with **`anr`** **Type:** object **Default:** color is "black"; length is 5; style is "point".

annotations.highContrast	For charts that support annotations (https://developers.google. **`annotations.highContrast`** boolean lets you override Goo color. By default, **`annotations.highContrast`** is true, which color with good contrast: light colors on dark backgrounds, and **`annotations.highContrast`** to false and don't specify your will use the default series color for the annotation:  **Type:** boolean **Default:** true
annotations.stem	For charts that support annotations (https://developers.google. **`annotations.stem`** object lets you override Google Charts' ch color with **`annotations.stem.color`** and the stem length wi that the stem length option has no effect on annotations with st annotations, the stem length is always the same as the text, anc stem extends across the entire chart. **Type:** object **Default:** color is "black"; length is 5 for domain annotations and
annotations.style	For charts that support annotations (https://developers.google. **`annotations.style`** option lets you override Google Charts' either **`'line'`** or **`'point'`**. **Type:** string **Default:** 'point'

annotations.textStyle	For charts that support annotations (https://developers.google. **`annotations.textStyle`** object controls the appearance of
	`var options = {`
	`  annotations: {`
	`    textStyle: {`
	`      fontName: 'Times-Roman',`
	`      fontSize: 18,`
	`      bold: true,`
	`      italic: true,`
	`      // The color of the text.`
	`      color: '#871b47',`
	`      // The color of the text outline.`
	`      auraColor: '#d799ae',`
	`      // The transparency of the text.`
	`      opacity: 0.8`
	`    }`

	`  }`
	`};`
	
	This option is currently supported for area, bar, column, combo,
	supported by the Annotation Chart
	(https://developers.google.com/chart/interactive/docs/gallery/
	
	Type: object
	Default: null

axisTitlesPosition	Where to place the axis titles, compared to the chart area. Supp • in - Draw the axis titles inside the chart area. • out - Draw the axis titles outside the chart area. • none - Omit the axis titles. **Type:** string **Default:** 'out'
backgroundColor	The background color for the main area of the chart. Can be eit example: `'red'` or `'#00cc00'`, or an object with the following **Type:** string or object **Default:** 'white'
backgroundColor.stroke	The color of the chart border, as an HTML color string. **Type:** string **Default:** '#666'
backgroundColor.strokeWidth	The border width, in pixels. **Type:** number **Default:** 0
backgroundColor.fill	The chart fill color, as an HTML color string. **Type:** string **Default:** 'white'
bar.groupWidth	The width of a group of bars, specified in either of these format

	• Pixels (e.g. 50). • Percentage of the available width for each group (e.g. '20%'), no space between them. **Type:** number or string **Default:** The golden ratio (http://en.wikipedia.org/wiki/Golden_r
bars	Whether the bars in a **Material** Bar Chart (https://developers.google.com/chart/interactive/docs/gallery/ horizontal. This option has no effect on Classic Bar Charts or Cl **Type:** 'horizontal' or 'vertical' **Default:** 'vertical'
chartArea	An object with members to configure the placement and size of drawn, excluding axis and legends). Two formats are supported A simple number is a value in pixels; a number followed by % is `{left:20,top:0,width:'50%',height:'75%'}` **Type:** object **Default:** null
chartArea.backgroundColor	Chart area background color. When a string is used, it can be eit English color name. When an object is used, the following prope • `stroke`: the color, provided as a hex string or English color r • `strokeWidth`: if provided, draws a border around the chart color of `stroke`). **Type:** string or object **Default:** 'white'
chartArea.left	How far to draw the chart from the left border. **Type:** number or string **Default:** auto
chartArea.top	How far to draw the chart from the top border. **Type:** number or string **Default:** auto
chartArea.width	Chart area width. **Type:** number or string **Default:** auto
chartArea.height	Chart area height. **Type:** number or string **Default:** auto

chart.subtitle	For **Material Charts** (https://developers.google.com/chart/intera this option specifies the subtitle. Only Material Charts support s **Type:** string **Default:** null
chart.title	For **Material Charts** (https://developers.google.com/chart/intera this option specifies the title. **Type:** string **Default:** null
colors	The colors to use for the chart elements. An array of strings, wh string, for example: `colors:['red','#004411']`. **Type:** Array of strings **Default:** default colors
dataOpacity	The transparency of data points, with 1.0 being completely opac histogram, bar, and column charts, this refers to the visible data rectangles in the others. In charts where *selecting data* creates this refers to the circles that appear upon hover or selection. Th and this option has no effect on other charts. (To change the op opacity (https://developers.google.com/chart/interactive/docs **Type:** number **Default:** 1.0
enableInteractivity	Whether the chart throws user-based events or reacts to user in throw 'select' or other interaction-based events (but *will* throw re hovertext or otherwise change depending on user input. **Type:** boolean **Default:** true
explorer	The `explorer` option allows users to pan and zoom Google ch default explorer behavior, enabling users to pan horizontally anc and out by scrolling. This feature is **experimental** and may change in future releases. ★ **Note:** The explorer only works with continuous axes (such as nu **Type:** object **Default:** null
explorer.actions	The Google Charts explorer supports three actions: • `dragToPan`: Drag to pan around the chart horizontally and v horizontal axis, use `explorer: { axis: 'horizontal'`

	• **`dragToZoom`**: The explorer's default behavior is to zoom in a **`explorer: { actions: ['dragToZoom', 'rightCl`** across a rectangular area zooms into that area. We recomm whenever **`dragToZoom`** is used. See **`explorer.maxZoomI`** **`explorer.zoomDelta`** for zoom customizations. • **`rightClickToReset`**: Right clicking on the chart returns it **Type:** Array of strings **Default:** ['dragToPan', 'rightClickToReset']
explorer.axis	By default, users can pan both horizontally and vertically when t want to users to only pan horizontally, use **`explorer: { axis`** **`explorer: { axis: 'vertical' }`** enables vertical-only p **Type:** string **Default:** both horizontal and vertical panning
explorer.keepInBounds	By default, users can pan all around, regardless of where the da beyond the original chart, use **`explorer: { keepInBounds:`** **Type:** boolean **Default:** false
explorer.maxZoomIn	The maximum that the explorer can zoom in. By default, users v they'll see only 25% of the original view. Setting **`explorer: {`** zoom in only far enough to see half of the original view. **Type:** number **Default:** 0.25
explorer.maxZoomOut	The maximum that the explorer can zoom out. By default, users that the chart will take up only 1/4 of the available space. Settin would let users zoom out far enough that the chart would take u **Type:** number **Default:** 4
explorer.zoomDelta	When users zoom in or out, **`explorer.zoomDelta`** determines the number, the smoother and slower the zoom. **Type:** number **Default:** 1.5
focusTarget	The type of the entity that receives focus on mouse hover. Also mouse click, and which data table element is associated with ev • 'datum' - Focus on a single data point. Correlates to a cell in t • 'category' - Focus on a grouping of all data points along the r data table.

	In focusTarget 'category' the tooltip displays all the category val values of different series. **Type:** string **Default:** 'datum'
fontSize	The default font size, in pixels, of all text in the chart. You can o chart elements. **Type:** number **Default:** automatic
fontName	The default font face for all text in the chart. You can override tl elements. **Type:** string **Default:** 'Arial'
forceIFrame	Draws the chart inside an inline frame. (Note that on IE8, this op in i-frames.) **Type:** boolean **Default:** false

hAxis	An object with members to configure various horizontal axis ele object, you can use object literal notation, as shown here:
	
	`{`
	`  title: 'Hello',`
	`  titleTextStyle: {`
	`    color: '#FF0000'`
	`  }`
	`}`
	
	Type: object
	Default: null

hAxis.baseline	The baseline for the horizontal axis. This option is only supported for a `continuous` (https://developers.google.com/chart/interactive/docs/custon **Type:** number **Default:** automatic
hAxis.baselineColor	The color of the baseline for the horizontal axis. Can be any HTI `'#00cc00'`. This option is only supported for a `continuous` (https://developers.google.com/chart/interactive/docs/custon

	Type: number **Default:** 'black'
hAxis.direction	The direction in which the values along the horizontal axis grow values. **Type:** 1 or -1 **Default:** 1
hAxis.format	A format string for numeric or date axis labels. For number axis labels, this is a subset of the decimal formattin (http://icu-project.org/apiref/icu4c/classDecimalFormat.html#. `{format:'#,###%'}` will display values "1,000%", "750%", and can also supply any of the following: • `{format: 'none'}`: displays numbers with no formatting • `{format: 'decimal'}`: displays numbers with thousands • `{format: 'scientific'}`: displays numbers in scientific • `{format: 'currency'}`: displays numbers in the local cu • `{format: 'percent'}`: displays numbers as percentages • `{format: 'short'}`: displays abbreviated numbers (e.g., • `{format: 'long'}`: displays numbers as full words (e.g., For date axis labels, this is a subset of the date formatting ICU (http://icu-project.org/apiref/icu4c/classSimpleDateFormat.htr `{format:'MMM d, y'}` will display the value "Jul 1, 2011" for The actual formatting applied to the label is derived from the lo more details, see loading charts with a specific locale (https://developers.google.com/chart/interactive/docs/library_ . This option is only supported for a `continuous` (https://developers.google.com/chart/interactive/docs/custom **Type:** string **Default:** auto
hAxis.gridlines	An object with members to configure the gridlines on the horizo object, you can use object literal notation, as shown here: `{color: '#333', count: 4}` This option is only supported for a `continuous` (https://developers.google.com/chart/interactive/docs/custom

	Type: object **Default:** null
hAxis.gridlines.color	The color of the horizontal gridlines inside the chart area. Speci **Type:** string **Default:** '#CCC'
hAxis.gridlines.count	The number of horizontal gridlines inside the chart area. Minimu automatically compute the number of gridlines. **Type:** number **Default:** 5

hAxis.gridlines.units	Overrides the default format for various aspects of date/datetin with chart computed gridlines. Allows formatting for years, mor milliseconds.
	
	General format is:
	
	`gridlines: {`
	`  units: {`
	`    years: {format: [/*format strings here*/]},`
	`    months: {format: [/*format strings here*/]}`
	`    days: {format: [/*format strings here*/]}`
	`    hours: {format: [/*format strings here*/]}`
	`    minutes: {format: [/*format strings here*/]`
	`    seconds: {format: [/*format strings here*/]`
	`    milliseconds: {format: [/*format strings he`
	`  }`
	`}`
	
	Additional information can be found in Dates and Times (https://developers.google.com/chart/interactive/docs/datesa
	
	Type: object
	Default: null

hAxis.minorGridlines	An object with members to configure the minor gridlines on the hAxis.gridlines option. This option is only supported for a `continuous` (https://developers.google.com/chart/interactive/docs/custom **Type:** object **Default:** null
hAxis.minorGridlines.color	The color of the horizontal minor gridlines inside the chart area.

	Type: string **Default:** A blend of the gridline and background colors
hAxis.minorGridlines.count	The number of horizontal minor gridlines between two regular g **Type:** number **Default:** 0
hAxis.minorGridlines.units	Overrides the default format for various aspects of date/datetin with chart computed minorGridlines. Allows formatting for year seconds, and milliseconds. General format is: `gridlines: {` `units: {` `years: {format: [/*format strings here*/]},` `months: {format: [/*format strings here*/]}` `days: {format: [/*format strings here*/]}` `hours: {format: [/*format strings here*/]}` `minutes: {format: [/*format strings here*/]` `seconds: {format: [/*format strings here*/]` `milliseconds: {format: [/*format strings he` `}` `}` Additional information can be found in Dates and Times (https://developers.google.com/chart/interactive/docs/datesa **Type:** object **Default:** null
hAxis.logScale	**`hAxis`** property that makes the horizontal axis a logarithmic sc Set to true for yes. This option is only supported for a `continuous` (https://developers.google.com/chart/interactive/docs/custom **Type:** boolean **Default:** false
hAxis.scaleType	**`hAxis`** property that makes the horizontal axis a logarithmic sc • null - No logarithmic scaling is performed. • 'log' - Logarithmic scaling. Negative and zero values are not setting **`hAxis: { logscale: true }`**. • 'mirrorLog' - Logarithmic scaling in which negative and zero a negative number is the negative of the log of the absolute

	linear scale. This option is only supported for a continuous (https://developers.google.com/chart/interactive/docs/custom **Type:** string **Default: null**
hAxis.textPosition	Position of the horizontal axis text, relative to the chart area. Su **Type:** string **Default:** 'out'

hAxis.textStyle	An object that specifies the horizontal axis text style. The object
	
	`{ color: <string>,`
	`  fontName: <string>,`
	`  fontSize: <number>,`
	`  bold: <boolean>,`
	`  italic: <boolean> }`
	
	The `color` can be any HTML color string, for example: `'red'` c `fontSize`.
	
	Type: object
	Default: `{color: 'black', fontName: <global-font-r size>}`

hAxis.ticks	Replaces the automatically generated X-axis ticks with the spec should be either a valid tick value (such as a number, date, datet an object, it should have a **v** property for the tick value, and an c string to be displayed as the label. Examples: • `hAxis: { ticks: [5,10,15,20] }` • `hAxis: { ticks: [{v:32, f:'thirty two'}, {v:(` • `hAxis: { ticks: [new Date(2014,3,15), new Da` • `hAxis: { ticks: [16, {v:32, f:'thirty two'}, }` This option is only supported for a continuous (https://developers.google.com/chart/interactive/docs/custom **Type:** Array of elements **Default:** auto
hAxis.title	`hAxis` property that specifies the title of the horizontal axis.

	Type: string **Default:** null

hAxis.titleTextStyle	An object that specifies the horizontal axis title text style. The o
	`{ color: <string>,`
	`  fontName: <string>,`
	`  fontSize: <number>,`
	`  bold: <boolean>,`
	`  italic: <boolean> }`
	The **`color`** can be any HTML color string, for example: **`'red'`** c **`fontSize`**.
	Type: object
	Default: **`{color: 'black', fontName: <global-font-r size>}`**

hAxis.allowContainerBoundaryTextCufoff	If false, will hide outermost labels rather than allow them to be c will allow label cropping. This option is only supported for a [discrete](https://developers.google.com/chart/interactive/docs/custom) (https://developers.google.com/chart/interactive/docs/custom **Type:** boolean **Default:** false
hAxis.slantedText	If true, draw the horizontal axis text at an angle, to help fit more horizontal axis text upright. Default behavior is to slant text if it that this option is available only when the **`hAxis.textPositi`** This option is only supported for a [discrete](https://developers.google.com/chart/interactive/docs/custom) (https://developers.google.com/chart/interactive/docs/custom **Type:** boolean **Default:** automatic
hAxis.slantedTextAngle	The angle of the horizontal axis text, if it's drawn slanted. Ignore is in auto mode, and the chart decided to draw the text horizont This option is only supported for a [discrete](https://developers.google.com/chart/interactive/docs/custom) (https://developers.google.com/chart/interactive/docs/custom **Type:** number, 1—90 **Default:** 30
hAxis.maxAlternation	Maximum number of levels of horizontal axis text. If axis text la might shift neighboring labels up or down in order to fit labels cl most number of levels to use; the server can use fewer levels, if

	This option is only supported for a discrete (https://developers.google.com/chart/interactive/docs/custom **Type:** number **Default:** 2
hAxis.maxTextLines	Maximum number of lines allowed for the text labels. Labels ca and the number of lines is, by default, limited by the height of th This option is only supported for a discrete (https://developers.google.com/chart/interactive/docs/custom **Type:** number **Default:** auto
hAxis.minTextSpacing	Minimum horizontal spacing, in pixels, allowed between two adj spaced too densely, or they are too long, the spacing can drop b of the label-unclutter measures will be applied (e.g, truncating tl This option is only supported for a discrete (https://developers.google.com/chart/interactive/docs/custom **Type:** number **Default:** The value of `hAxis.textStyle.fontSize`
hAxis.showTextEvery	How many horizontal axis labels to show, where 1 means show label, and so on. Default is to try to show as many labels as pos This option is only supported for a discrete (https://developers.google.com/chart/interactive/docs/custom **Type:** number **Default:** automatic
hAxis.maxValue	Moves the max value of the horizontal axis to the specified valu Ignored if this is set to a value smaller than the maximum x-valu `hAxis.viewWindow.max` overrides this property. This option is only supported for a continuous (https://developers.google.com/chart/interactive/docs/custom **Type:** number **Default:** automatic
hAxis.minValue	Moves the min value of the horizontal axis to the specified value Ignored if this is set to a value greater than the minimum x-value `hAxis.viewWindow.min` overrides this property. This option is only supported for a continuous (https://developers.google.com/chart/interactive/docs/custom **Type:** number

	Default: automatic
hAxis.viewWindowMode	Specifies how to scale the horizontal axis to render the values w string values are supported: • 'pretty' - Scale the horizontal values so that the maximum an bit inside the left and right of the chart area. This will cause l **haxis.viewWindow.max** to be ignored. • 'maximized' - Scale the horizontal values so that the maximu left and right of the chart area. This will cause **haxis.view** **haxis.viewWindow.max** to be ignored. • 'explicit' - A deprecated option for specifying the left and righ (Deprecated because it's redundant with **haxis.viewWind** **haxis.viewWindow.max**.) Data values outside these value **hAxis.viewWindow** object describing the maximum and m This option is only supported for a continuous (https://developers.google.com/chart/interactive/docs/custom **Type:** string **Default:** Equivalent to 'pretty', but **haxis.viewWindow.min** an precedence if used.
hAxis.viewWindow	Specifies the cropping range of the horizontal axis. **Type:** object **Default:** null
hAxis.viewWindow.max	• For a continuous (https://developers.google.com/chart/interactive/docs/cus The maximum horizontal data value to render. • For a discrete (https://developers.google.com/chart/interactive/docs/cus The zero-based row index where the cropping window ends. be cropped out. In conjunction with **vAxis.viewWindowMo** [min, max) that denotes the element indices to display. In oth **index < max** will be displayed. Ignored when **hAxis.viewWindowMode** is 'pretty' or 'maximize **Type:** number **Default:** auto
hAxis.viewWindow.min	• For a continuous (https://developers.google.com/chart/interactive/docs/cus The minimum horizontal data value to render. • For a discrete (https://developers.google.com/chart/interactive/docs/cus

	The zero-based row index where the cropping window begins will be cropped out. In conjunction with **`vAxis.viewWindo`** range [min, max) that denotes the element indices to display **`min <= index < max`** will be displayed. Ignored when **`hAxis.viewWindowMode`** is 'pretty' or 'maximize **Type:** number **Default:** auto
height	Height of the chart, in pixels. **Type:** number **Default:** height of the containing element
isStacked	If set to true, stacks the elements for all series at each domain v (https://developers.google.com/chart/interactive/docs/gallery/ (https://developers.google.com/chart/interactive/docs/gallery/ (https://developers.google.com/chart/interactive/docs/gallery/ Charts reverses the order of legend items to better correspond v (E.g. series 0 will be the bottom-most legend item). This **does n** (https://developers.google.com/chart/interactive/docs/gallery/ The **`isStacked`** option also supports 100% stacking, where the value are rescaled to add up to 100%. The options for **`isStacked`** are: • **`false`** — elements will not stack. This is the default option. • **`true`** — stacks elements for all series at each domain value. • **`'percent'`** — stacks elements for all series at each domain add up to 100%, with each element's value calculated as a pe • **`'relative'`** — stacks elements for all series at each doma they add up to 1, with each element's value calculated as a fr • **`'absolute'`** — functions the same as **`isStacked: true`**. For 100% stacking, the calculated value for each element will ap The target axis will default to tick values based on the relative 0 **`'relative'`**, and 0-100% for **`'percent'`** (**Note:** when using th values are displayed as percentages, however the actual values because the percentage axis ticks are the result of applying a fo values. When using **`isStacked: 'percent'`**, be sure to spec 0-1 scale values). You can customize the gridlines/tick values a **`hAxis/vAxis`** options. 100% stacking only supports data values of type **`number`**, and n **Type:** boolean/string

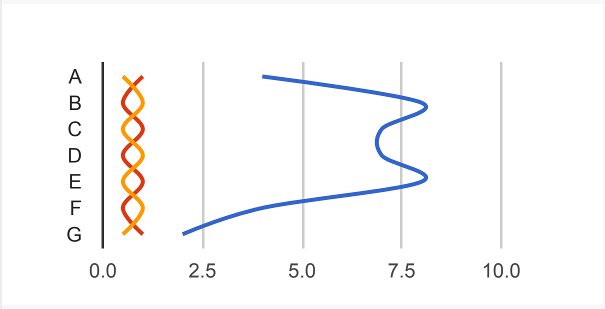
	Default: false
legend	An object with members to configure various aspects of the leg you can use object literal notation, as shown here: `{position: 'top', textStyle: {color: 'blue', fc` **Type:** object **Default:** null
legend.position	Position of the legend. Can be one of the following: • 'bottom' - Below the chart. • 'left' - To the left of the chart, provided the left axis has no se the legend on the left, use the option **`targetAxisIndex:`** • 'in' - Inside the chart, by the top left corner. • 'none' - No legend is displayed. • 'right' - To the right of the chart. Incompatible with the **`vAxes`** • 'top' - Above the chart. **Type:** string **Default:** 'right'
legend.alignment	Alignment of the legend. Can be one of the following: • 'start' - Aligned to the start of the area allocated for the leger • 'center' - Centered in the area allocated for the legend. • 'end' - Aligned to the end of the area allocated for the legend. Start, center, and end are relative to the style -- vertical or horizo 'right' legend, 'start' and 'end' are at the top and bottom, respecti would be at the left and right of the area, respectively. The default value depends on the legend's position. For 'bottom legends default to 'start'. **Type:** string **Default:** automatic
legend.textStyle	An object that specifies the legend text style. The object has thi `{ color: <string>, fontName: <string>, fontSize: <number>, bold: <boolean>, italic: <boolean> }`

The `color` can be any HTML color string, for example: `'red'` c `fontSize`.

Type: object
Default: `{color: 'black', fontName: <global-font-r size>}`

orientation

The orientation of the chart. When set to `'vertical'`, rotates instance) a column chart becomes a bar chart, and an area cha



Type: string
Default: 'horizontal'

reverseCategories

If set to true, will draw series from right to left. The default is to

This option is only supported for a [discrete](https://developers.google.com/chart/interactive/docs/custom
(https://developers.google.com/chart/interactive/docs/custom
(https://developers.google.com/chart/interactive/docs/custom

Type: boolean
Default: false

series

An array of objects, each describing the format of the correspor values for a series, specify an empty object {}. If a series or a va be used. Each object supports the following properties:

- `annotations` - An object to be applied to annotations for tl for instance, the `textStyle` for the series:

  ```
  series: {
    0: {
      annotations: {
        textStyle: {fontSize: 12, color: 'red'
      }
    }
  }
  ```

  See the various `annotations` options for a more complete
- `color` - The color to use for this series. Specify a valid HTM

- **`targetAxisIndex`** - Which axis to assign this series to, wh
  opposite axis. Default value is 0; set to 1 to define a chart wh
  against different axes. At least one series much be allocated
  different scale for different axes.
- **`visibleInLegend`** - A boolean value, where true means tha
  and false means that it should not. Default is true.

You can specify either an array of objects, each of which applies
can specify an object where each child has a numeric key indica
example, the following two declarations are identical, and decla
from the legend, and the fourth as red and absent from the lege

```
series: [
  {color: 'black', visibleInLegend: false}, {},
  {color: 'red', visibleInLegend: false}
]
series: {
  0:{color: 'black', visibleInLegend: false},
  3:{color: 'red', visibleInLegend: false}
}
```

Type: Array of objects, or object with nested objects
Default: {}

theme

A theme is a set of predefined option values that work together
visual effect. Currently only one theme is available:

- 'maximized' - Maximizes the area of the chart, and draws the
  chart area. Sets the following options:

  ```
  chartArea: {width: '100%', height: '100%'},
  legend: {position: 'in'},
  titlePosition: 'in', axisTitlesPosition: 'in
  hAxis: {textPosition: 'in'}, vAxis: {textPos
  ```

Type: string
Default: null

title

Text to display above the chart.

Type: string
Default: no title

titlePosition

Where to place the chart title, compared to the chart area. Supp

- in - Draw the title inside the chart area.

	• out - Draw the title outside the chart area. • none - Omit the title. **Type:** string **Default:** 'out'

titleTextStyle	An object that specifies the title text style. The object has this fo
	`{ color: <string>,`
	`  fontName: <string>,`
	`  fontSize: <number>,`
	`  bold: <boolean>,`
	`  italic: <boolean> }`
	The **`color`** can be any HTML color string, for example: **`'red'`** c **`fontSize`**.
	Type: object
	Default: **`{color: 'black', fontName: <global-font-r size>}`**

tooltip	An object with members to configure various tooltip elements. can use object literal notation, as shown here: `{textStyle: {color: '#FF0000'}, showColorCode:` **Type:** object **Default:** null
tooltip.ignoreBounds	If set to **`true`**, allows the drawing of tooltips to flow outside of t **Note:** This only applies to HTML tooltips. If this is enabled with the chart bounds will be cropped. See Customizing Tooltip Cont (https://developers.google.com/chart/interactive/docs/custom **Type:** boolean **Default:** false
tooltip.isHtml	If set to true, use HTML-rendered (rather than SVG-rendered) too (https://developers.google.com/chart/interactive/docs/custom **Note:** customization of the HTML tooltip content via the tooltip (https://developers.google.com/chart/interactive/docs/roles#t Bubble Chart (https://developers.google.com/chart/interactive/ **Type:** boolean **Default:** false

tooltip.showColorCode	If true, show colored squares next to the series information in th **`focusTarget`** is set to 'category', otherwise the default is false **Type:** boolean **Default:** automatic

tooltip.textStyle	An object that specifies the tooltip text style. The object has this
	
	`{ color: <string>,`
	`  fontName: <string>,`
	`  fontSize: <number>,`
	`  bold: <boolean>,`
	`  italic: <boolean> }`
	
	The **`color`** can be any HTML color string, for example: **`'red'`** c **`fontSize`**.
	
	Type: object
	Default: **`{color: 'black', fontName: <global-font-r size>}`**

tooltip.trigger	The user interaction that causes the tooltip to be displayed: • 'focus' - The tooltip will be displayed when the user hovers ov • 'none' - The tooltip will not be displayed. • 'selection' - The tooltip will be displayed when the user select **Type:** string **Default:** 'focus'

trendlines	Displays trendlines (https://developers.google.com/chart/intera charts that support them. By default, linear trendlines are used, **`trendlines.`*`n`*`.type`** option.
	
	Trendlines are specified on a per-series basis, so most of the tir
	
	`var options = {`
	`  trendlines: {`
	`    0: {`
	`      type: 'linear',`
	`      color: 'green',`
	`      lineWidth: 3,`
	`      opacity: 0.3,`
	`      showR2: true,`
	`      visibleInLegend: true`
	`    }`
	`  }`

	} **Type:** object **Default:** null
trendlines.n.color	The color of the trendline (https://developers.google.com/char expressed as either an English color name or a hex string. **Type:** string **Default:** default series color
trendlines.n.degree	For trendlines (https://developers.google.com/chart/interactive `'polynomial'`, the degree of the polynomial (**2** for quadratic, degree may change from 3 to 2 in an upcoming release of Goog **Type:** number **Default:** 3
trendlines.n.labelInLegend	If set, the trendline (https://developers.google.com/chart/intera appear in the legend as this string. **Type:** string **Default:** null
trendlines.n.lineWidth	The line width of the trendline (https://developers.google.com/chart/interactive/docs/gallery, **Type:** number **Default:** 2
trendlines.n.opacity	The transparency of the trendline (https://developers.google.com/chart/interactive/docs/gallery, 1.0 (opaque). **Type:** number **Default:** 1.0
trendlines.n.pointSize	Trendlines (https://developers.google.com/chart/interactive/do by stamping a bunch of dots on the chart; this rarely-needed op dots. The trendline's `lineWidth` option will usually be preferab you're using the global `pointSize` option and want a different **Type:** number **Default:** 1
trendlines.n.pointsVisible	Trendlines (https://developers.google.com/chart/interactive/do by stamping a bunch of dots on the chart. The trendline's `poin` the points for a particular trendline are visible. **Type:** boolean

	Default: true
trendlines.n.showR2	Whether to show the coefficient of determination (https://developers.google.com/chart/interactive/docs/gallery/ tooltip. **Type:** boolean **Default:** false
trendlines.n.type	Whether the trendlines (https://developers.google.com/chart/i `'linear'` (the default), `'exponential'`, or `'polynomial'`. **Type:** string **Default:** linear
trendlines.n.visibleInLegend	Whether the trendline (https://developers.google.com/chart/int equation appears in the legend. (It will appear in the trendline to **Type:** boolean **Default:** false
vAxes	Specifies properties for individual vertical axes, if the chart has is a `vAxis` object, and can contain all the properties supported override any global settings for the same property. To specify a chart with multiple vertical axes, first define a new `series.targetAxisIndex`, then configure the axis using `vA` series 2 to the right axis and specifies a custom title and text st

```
{
  series: {
    2: {
      targetAxisIndex:1
    }
  },
  vAxes: {
    1: {
      title:'Losses',
      textStyle: {color: 'red'}
    }
  }
}
```

This property can be either an object or an array: the object is a numeric label that specifies the axis that it defines--this is the fo of objects, one per axis. For example, the following array-style n shown above:

	`vAxes: [`
	`  {}, // Nothing specified for axis 0`
	`  {`
	`    title:'Losses',`
	`    textStyle: {color: 'red'} // Axis 1`
	`  }`
	`]`
	
	Type: Array of object, or object with child objects
	Default: null

vAxis	An object with members to configure various vertical axis eleme you can use object literal notation, as shown here: `{title: 'Hello', titleTextStyle: {color: '#FF0` **Type:** object **Default:** null
vAxis.baseline	**`vAxis`** property that specifies the baseline for the vertical axis. grid line or smaller than the lowest grid line, it will be rounded tc **Type:** number **Default:** automatic
vAxis.baselineColor	Specifies the color of the baseline for the vertical axis. Can be a **`'red'`** or **`'#00cc00'`**. **Type:** number **Default:** 'black'
vAxis.direction	The direction in which the values along the vertical axis grow. S values. **Type:** 1 or -1 **Default:** 1
vAxis.format	A format string for numeric axis labels. This is a subset of the I (http://icu-project.org/apiref/icu4c/classDecimalFormat.html# **`{format:'#,###%'}`** will display values "1,000%", "750%", and can also supply any of the following: • **`{format: 'none'}`**: displays numbers with no formatting • **`{format: 'decimal'}`**: displays numbers with thousands • **`{format: 'scientific'}`**: displays numbers in scientific • **`{format: 'currency'}`**: displays numbers in the local cu • **`{format: 'percent'}`**: displays numbers as percentages

	• `{format: 'short'}`: displays abbreviated numbers (e.g., • `{format: 'long'}`: displays numbers as full words (e.g., The actual formatting applied to the label is derived from the loc more details, see loading charts with a specific locale (https://developers.google.com/chart/interactive/docs/library_ . **Type:** string **Default:** auto
vAxis.gridlines	An object with members to configure the gridlines on the vertica object, you can use object literal notation, as shown here: `{color: '#333', count: 4}` **Type:** object **Default:** null
vAxis.gridlines.color	The color of the vertical gridlines inside the chart area. Specify a **Type:** string **Default:** '#CCC'
vAxis.gridlines.count	The number of vertical gridlines inside the chart area. Minimum compute the number of gridlines. **Type:** number **Default:** 5

vAxis.gridlines.units	Overrides the default format for various aspects of date/datetin with chart computed gridlines. Allows formatting for years, mor milliseconds.
	
	General format is:
	
	`gridlines: {`
	`  units: {`
	`    years: {format: [/*format strings here*/]},`
	`    months: {format: [/*format strings here*/]}`
	`    days: {format: [/*format strings here*/]}`
	`    hours: {format: [/*format strings here*/]}`
	`    minutes: {format: [/*format strings here*/]`
	`    seconds: {format: [/*format strings here*/]`
	`    milliseconds: {format: [/*format strings he`
	`  }`
	`}`

	Additional information can be found in Dates and Times (https://developers.google.com/chart/interactive/docs/datesa **Type:** object **Default:** null
vAxis.minorGridlines	An object with members to configure the minor gridlines on the vAxis.gridlines option. **Type:** object **Default:** null
vAxis.minorGridlines.color	The color of the vertical minor gridlines inside the chart area. Sp **Type:** string **Default:** A blend of the gridline and background colors
vAxis.minorGridlines.count	The number of vertical minor gridlines between two regular grid **Type:** number **Default:** 0
vAxis.minorGridlines.units	Overrides the default format for various aspects of date/datetin with chart computed minorGridlines. Allows formatting for year seconds, and milliseconds. General format is:

```
gridlines: {
  units: {
    years: {format: [/*format strings here*/]},
    months: {format: [/*format strings here*/]}
    days: {format: [/*format strings here*/]}
    hours: {format: [/*format strings here*/]}
    minutes: {format: [/*format strings here*/]
    seconds: {format: [/*format strings here*/]
    milliseconds: {format: [/*format strings he
  }
}
```

	Additional information can be found in Dates and Times (https://developers.google.com/chart/interactive/docs/datesa **Type:** object **Default:** null
vAxis.logScale	If true, makes the vertical axis a logarithmic scale. Note: All valu **Type:** boolean

	Default: false
vAxis.scaleType	`vAxis` property that makes the vertical axis a logarithmic scale • null - No logarithmic scaling is performed. • 'log' - Logarithmic scaling. Negative and zero values are not setting `vAxis: { logscale: true }`. • 'mirrorLog' - Logarithmic scaling in which negative and zero a negative number is the negative of the log of the absolute linear scale. This option is only supported for a [continuous](https://developers.google.com/chart/interactive/docs/custom) (https://developers.google.com/chart/interactive/docs/custom **Type:** string **Default:** null
vAxis.textPosition	Position of the vertical axis text, relative to the chart area. Supp **Type:** string **Default:** 'out'

vAxis.textStyle	An object that specifies the vertical axis text style. The object ha
	
	`{ color: <string>,`
	`  fontName: <string>,`
	`  fontSize: <number>,`
	`  bold: <boolean>,`
	`  italic: <boolean> }`
	
	The `color` can be any HTML color string, for example: `'red'` c
	`fontSize`.
	
	Type: object
	Default: `{color: 'black', fontName: <global-font-r`
	`size>}`

vAxis.ticks	Replaces the automatically generated Y-axis ticks with the spec should be either a valid tick value (such as a number, date, datet an object, it should have a `v` property for the tick value, and an c string to be displayed as the label. Examples: • `vAxis: { ticks: [5,10,15,20] }` • `vAxis: { ticks: [{v:32, f:'thirty two'}, {v:(` • `vAxis: { ticks: [new Date(2014,3,15), new Da`

	• `vAxis: { ticks: [16, {v:32, f:'thirty two'}, }` **Type:** Array of elements **Default:** auto
vAxis.title	`vAxis` property that specifies a title for the vertical axis. **Type:** string **Default:** no title
vAxis.titleTextStyle	An object that specifies the vertical axis title text style. The obje `{ color: <string>, fontName: <string>, fontSize: <number>, bold: <boolean>, italic: <boolean> }` The `color` can be any HTML color string, for example: `'red'` c `fontSize`. **Type:** object **Default:** `{color: 'black', fontName: <global-font-r size>}`
vAxis.maxValue	Moves the max value of the vertical axis to the specified value; t Ignored if this is set to a value smaller than the maximum y-valu `vAxis.viewWindow.max` overrides this property. **Type:** number **Default:** automatic
vAxis.minValue	Moves the min value of the vertical axis to the specified value; t Ignored if this is set to a value greater than the minimum y-value `vAxis.viewWindow.min` overrides this property. **Type:** number **Default:** null
vAxis.viewWindowMode	Specifies how to scale the vertical axis to render the values with values are supported: • 'pretty' - Scale the vertical values so that the maximum and r inside the top and bottom of the chart area. This will cause v `vaxis.viewWindow.max` to be ignored. • 'maximized' - Scale the vertical values so that the maximum and bottom of the chart area. This will cause `vaxis.viewW` `vaxis.viewWindow.max` to be ignored.

	• 'explicit' - A deprecated option for specifying the top and bot (Deprecated because it's redundant with **vaxis.viewWind** **vaxis.viewWindow.max**. Data values outside these value **vAxis.viewWindow** object describing the maximum and m **Type:** string **Default:** Equivalent to 'pretty', but **vaxis.viewWindow.min** an precedence if used.
vAxis.viewWindow	Specifies the cropping range of the vertical axis. **Type:** object **Default:** null
vAxis.viewWindow.max	The maximum vertical data value to render. Ignored when **vAxis.viewWindowMode** is 'pretty' or 'maximize **Type:** number **Default:** auto
vAxis.viewWindow.min	The minimum horizontal data value to render. Ignored when **vAxis.viewWindowMode** is 'pretty' or 'maximize **Type:** number **Default:** auto
width	Width of the chart, in pixels. **Type:** number **Default:** width of the containing element

Methods

Method	
`draw(data, options)`	Draws the chart. The chart accepts further method calls only after the (#Events)event is fired. [Extended description](https://developers.google.com/chart/interactive/docs/reference#vis) (https://developers.google.com/chart/interactive/docs/reference#vis **Return Type:** none
`getAction(actionID)`	Returns the tooltip action object with the requested **actionID**. **Return Type:** object
`getBoundingBox(id)`	Returns an object containing the left, top, width, and height of chart ele

The format for **`id`** isn't yet documented (they're the return values of eve (https://developers.google.com/chart/interactive/docs/events)), but h some examples:

```
var cli = chart.getChartLayoutInterface();
```

Height of the chart area

```
    cli.getBoundingBox('chartarea').height
```

Width of the third bar in the first series of a bar or column cl

```
    cli.getBoundingBox('bar#0#2').width
```

Bounding box of the fifth wedge of a pie chart

```
    cli.getBoundingBox('slice#4')
```

Bounding box of the chart data of a vertical (e.g., column) c

```
    cli.getBoundingBox('vAxis#0#gridline')
```

Bounding box of the chart data of a horizontal (e.g., bar) cha

```
    cli.getBoundingBox('hAxis#0#gridline')
```

Values are relative to the container of the chart. Call this *after* the chart

Return Type: object

`getChartAreaBoundingBox()`

Returns an object containing the left, top, width, and height of the chart (i.e., excluding labels and legend):

```
var cli = chart.getChartLayoutInterface();

    cli.getChartAreaBoundingBox().left
    cli.getChartAreaBoundingBox().top
    cli.getChartAreaBoundingBox().height
    cli.getChartAreaBoundingBox().width
```

Values are relative to the container of the chart. Call this *after* the chart

Return Type: object

`getChartLayoutInterface()`	Returns an object containing information about the onscreen placeme chart and its elements. The following methods can be called on the returned object: • `getBoundingBox` • `getChartAreaBoundingBox` • `getHAxisValue` • `getVAxisValue` • `getXLocation` • `getYLocation` Call this *after* the chart is drawn. **Return Type:** object
`getHAxisValue(position, optional_axis_index)`	Returns the logical horizontal value at `position`, which is an offset fr container's left edge. Can be negative. Example: `chart.getChartLayoutInterface().getHAxisValue` Call this *after* the chart is drawn. **Return Type:** number
`getImageURI()`	Returns the chart serialized as an image URI. Call this *after* the chart is drawn. See Printing PNG Charts (https://developers.google.com/chart/interactive/docs/printing). **Return Type:** string
`getSelection()`	Returns an array of the selected chart entities. Selectable entities are b entries and categories. A bar corresponds to a cell in the data table, a l to a column (row index is null), and a category to a row (column index i this chart, only one entity can be selected at any given moment. [Exter description](https://developers.google.com/chart/interactive/docs/reference#vis) (https://developers.google.com/chart/interactive/docs/reference#vis . **Return Type:** Array of selection elements
`getVAxisValue(position, optional_axis_index)`	Returns the logical vertical value at `position`, which is an offset from container's top edge. Can be negative. Example: `chart.getChartLayoutInterface().getVAxisValue` Call this *after* the chart is drawn.

	Return Type: number
`getXLocation(position, optional_axis_index)`	Returns the screen x-coordinate of `position` relative to the chart's co Example: `chart.getChartLayoutInterface().getXLocation(` Call this *after* the chart is drawn. **Return Type:** number
`getYLocation(position, optional_axis_index)`	Returns the screen y-coordinate of `position` relative to the chart's co Example: `chart.getChartLayoutInterface().getYLocation(` Call this *after* the chart is drawn. **Return Type:** number
`removeAction(actionID)`	Removes the tooltip action with the requested `actionID` from the cha **Return Type: none**
`setAction(action)`	Sets a tooltip action to be executed when the user clicks on the action The `setAction` method takes an object as its action parameter. This should specify 3 properties: `id`— the ID of the action being set, `text` – should appear in the tooltip for the action, and `action` — the function be run when a user clicks on the action text. Any and all tooltip actions should be set prior to calling the chart's `dra` method. Extended description (https://developers.google.com/chart/interactive/docs/reference#vis **Return Type: none**
`setSelection()`	Selects the specified chart entities. Cancels any previous selection. Se entities are bars, legend entries and categories. A bar corresponds to a data table, a legend entry to a column (row index is null), and a categor (column index is null). For this chart, only one entity can be selected at `Extended description` (https://developers.google.com/chart/interactive/docs/reference#vis . **Return Type:** none
`clearChart()`	Clears the chart, and releases all of its allocated resources. **Return Type:** none

Events

For more information on how to use these events, see Basic Interactivity (https://developers.google.com/chart/interactive/docs/basic_interactivity), Handling Events (https://developers.google.com/chart/interactive/docs/events), and Firing Events (https://developers.google.com/chart/interactive/docs/dev/events).

Name	
`animationfinish`	Fired when transition animation is complete. **Properties:** none
`click`	Fired when the user clicks inside the chart. Can be used to identify when the title, data elements, legend entries, axes, gridlines, or labels are clicked. **Properties:** targetID
`error`	Fired when an error occurs when attempting to render the chart. **Properties:** id, message
`onmouseover`	Fired when the user mouses over a visual entity. Passes back the row and column indices of the corresponding data table element. **Properties:** row, column
`onmouseout`	Fired when the user mouses away from a visual entity. Passes back the row and column indices of the corresponding data table element. **Properties:** row, column
`ready`	The chart is ready for external method calls. If you want to interact with the chart, and call methods after you draw it, you should set up a listener for this event *before* you call the `draw` method, and call them only after the event was fired. **Properties:** none
`select`	Fired when the user clicks a visual entity. To learn what has been selected, call `getSelection()` (#Methods). **Properties:** none

Data policy

All code and data are processed and rendered in the browser. No data is sent to any server.

上次更新日期：二月 23, 2017