Line Charts

This document describes the various types of line charts that you can create using the Chart API.

Table of Contents

Chart Types (`cht`)
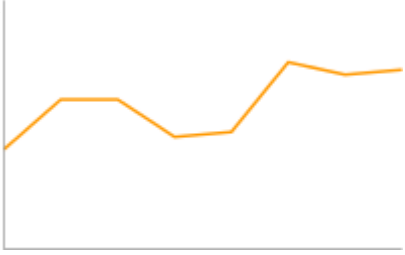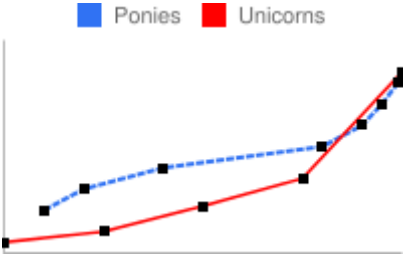
There are a variety of line charts that you can create. Specify a line chart with the following syntax:

Syntax

```
cht=<chart_type>
```

Where **<chart_type>** is one of the the following types:

Parameter	Description	Example

`lc`	A line chart where data points are spaced evenly along the *x*-axis. Axis lines are shown by default. To show values on the axis lines, you must specify chxt (#axis_type).	
		`cht=lc`
		`chd=t:40,60,60,45,47,75,70,72`

`ls`	Similar to `lc` charts, but by default does not display axis lines. These are also known as *sparklines*.	
		`cht=ls`
		`chd=t:27,25,60,31,25,39,25,`
		`31,26,28,80,28,27,31,27,`
		`29,26,35,70,25`

`lxy`	Lets you specify both x- and y-coordinates for each point, rather just the y values.
	
	To specify data points for `lxy` charts, you must specify series in multiples of two. The first series in each pair is the x coordinates for that line, and the second series is the corresponding y coordinates for that line. For example:
	
	`chd=t:<line_1_x1>,<line_1_x2>,...|<line_1_y1>,<line_1_y2>,...|`
	`    <line_2_x1>,<line_2_x2>,...|<line_2_y1>,<line_2_y2>,...`
	
	To space the data points evenly along the *x*-axis *only*, provide a single undefined value for the x set for that series. In this example, the red "Unicorns" line spaces the data points evenly along the x-axis. For more information about undefined values, see Data formats (https://developers.google.com/chart/image/docs/data_formats?hl=es).
	
	In this example, the "Ponies" line specifies exact x- and y-values for all points, and the "Unicorns" line spaces the values evenly along the x-axis.
		
		`cht=lxy`
		`chd=t:`
		`  10,20,40,80,90,95,99|`
		`  20,30,40,50,60,70,80|`
		`  -1|`
		`  5,10,22,35,85`

> ⭐ **Note:** If you are creating a [compound **1xy** chart](https://developers.google.com/chart/image/docs/gallery/compound_charts?hl=es), don't forget that you must show your data in multiples of two in the **chd** parameter. For example: `chd=t0:` to hide all lines, `chd=t2:` to draw one line, `chd=t4:` to draw two lines, and so on. If you specify `chd=t1:`, no lines will be drawn; if you specify `chd=t3:`, only one line will be drawn.
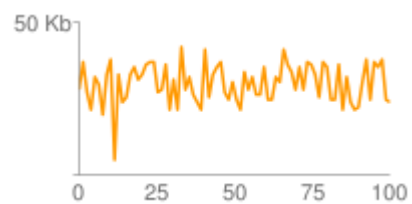
Back to top (#top)

Data Granularity

Take care not to overestimate the number of data points required for your line chart. For example, to show how popular chocolate ice cream was over the last ten years, aggregating search queries for each day would result in more than 3,600 values. It would not make any sense to plot a graph at this granularity. On a chart 1024 pixels wide, one data point would be about a quarter of a pixel. (Additionally, this would be too much data to pass in a URL). The following examples illustrate this point.
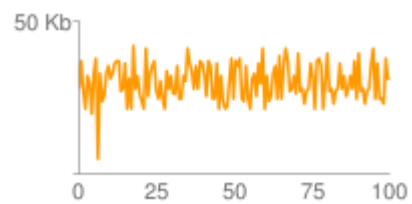
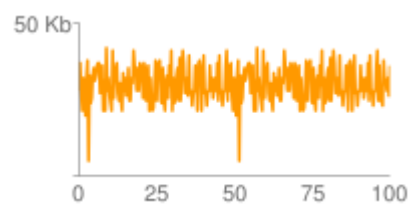200 pixel wide chart with 40 data points (5 pixels per data point):

80 data points (only 2.5 pixels per data point):

150 data points (only 1.3 pixels per data point):

300 data points (less than 1 pixel per data point):

Back to top (#top)

Series Colors `chco`

You can specify the colors of a specific series, or all series, using the `chco` parameter.

Syntax

```
chco=
<color_1>, ... <color_n>
```

<color>

> An [RRGGBB format hexadecimal number](https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). Specify a single value to apply the same color to all series. Specify different colors for different series by adding color values separated by a
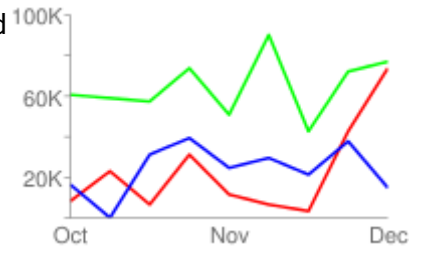
comma. If you have fewer colors than you have series, the unspecified series will cycle through the specified colors from the beginning.

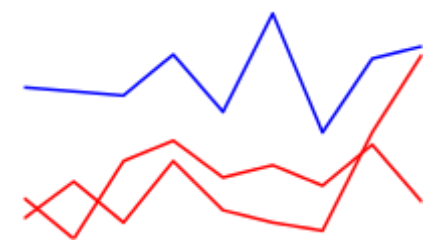**Examples**

Description	Example
When you specify a single color for each series, each series gets the color assigned. This example has three data series and three colors specified.	`chco=FF0000,00FF00,0000FF`
This example also has three data series, but only two colors are specified. Because the color for the third series is unspecified, the third line is drawn using the first color (red).	`chco=FF0000,0000FF`

Back to top (#top)

Compound Charts

You can add lines, candlesticks, and shape markers to a line chart to make a compound chart. See Compound Charts (https://developers.google.com/chart/image/docs/gallery/compound_charts?hl=es) for more details.

Standard Features

The rest of the features on this page are standard chart features.

Chart Title `chtt`, `chts` [*All charts*]

You can specify the title text, color, and font size for your chart.

Syntax

```
chtt=<chart_title>
chts=<color>,<font_size>,<opt_alignment>
```

`chtt` - Specifies the chart title.

<chart_title>

> Title to show for the chart. You cannot specify where this appears, but you can optionally specify the font size and color. Use a + sign to indicate spaces, and a pipe character (|) to indicate line breaks.

`chts` [*Optional*] - Colors and font size for the `chtt` parameter.

<color>

The title color, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). *Default color is black.*

<font_size>

Font size of the title, in points.

<opt_alignment>

[*Optional*] Alignment of the title. Choose one of the following case-sensitive string values: "l" (left), "c" (centered) "r" (right). *Default is "c".*

Examples

Description	Example

A chart with a title, using default color and font size.
 	
Specify a space with a plus sign (**+**).
 	
Use a pipe character (**|**) to force a line break.
 	
`chts` is not specified here.
	Site visitors by month
	January to July
	
	`chtt=Site+visitors+by+month|`
	`January+to+July`

A chart with a blue, right-aligned, 20-point title.	Site visitors
	
	`chtt=Site+visitors`
	`chts=FF0000,20,r`

Back to top (#top)

Chart Legend Text and Style `chdl`, `chdlp`, `chdls` [*All charts*]

The legend is a side section of the chart that gives a small text description of each series. You can specify the text associated with each series in this legend, and specify where on the chart it should appear.

See also chma (#gcharts_chart_margins), to learn how to set the margins around your legend.

A note on string values: Only URL-safe characters are permitted in label strings. To be safe, you should URL-encode any strings containing characters not in the character set `0-9a-zA-Z`. You can find a URL encoder in the Google Visualization Documentation (https://developers.google.com/chart/interactive/docs/querylanguage?hl=es#plainText).

Syntax

```
chdl=<data_series_1_label>|...|<data_series_n_label>
chdlp=<opt_position>|<opt_label_order>
chdls=<color>,<size>
```

`chdl` - The text for each series, to display in the legend.

<data_series_label>

The text for the legend entries. Each label applies to the corresponding series in the `chd` array. Use a + mark for a space. If you do not specify this parameter, the chart will not get a legend. There is no way to specify a line break in a label. The legend will typically expand to hold your legend text, and the chart area will shrink to accommodate the legend.

`chdlp` - [*Optional*] The position of the legend, and order of the legend entries. You can specify *<position>* and/or *<label_order>*. If you specify both, separate them with a bar character. You can add an 's' to any value if you want empty legend entries in `chdl` to be skipped in the legend. Examples: `chdlp=bv`, `chdlp=r`, `chdlp=bv|r`, `chdlp=bvs|r`

<*opt_position*>

[*Optional*] Specifies the position of the legend on the chart. To specify additional padding between the legend and the chart area or the image border, use the chma (#gcharts_chart_margins) parameter. Choose one of the following values:

- `b` - Legend at the bottom of the chart, legend entries in a horizontal row.
- `bv` - Legend at the bottom of the chart, legend entries in a vertical column.
- `t` - Legend at the top of the chart, legend entries in a horizontal row.
- `tv` - Legend at the top of the chart, legend entries in a vertical column.
- `r` - [*Default*] Legend to the right of the chart, legend entries in a vertical column.
- `l` - Legend to the left of the chart, legend entries in a vertical column.

<*opt_label_order*>

[*Optional*] The order in which the labels are shown in the legend. Choose one of the following value:

- `l` - [*Default for vertical legends*] Display labels in the order given to `chdl`.
- `r` - Display labels in the reverse order as given to `chdl`. This is useful in stacked bar charts to show the legend in the same order as the bars appear.
- `a` - [*Default for horizontal legends*] Automatic ordering: roughly means sorting by length, shortest first, as measured in 10 pixel blocks. When two elements are the same length (divided into 10 pixel blocks), the one listed first will appear first.
- `0,1,2...` - Custom label order. This is a list of zero-based label indexes from `chdl`, separated by commas.

`chdls` - [*Optional*] Specifies the color and font size of the legend text.
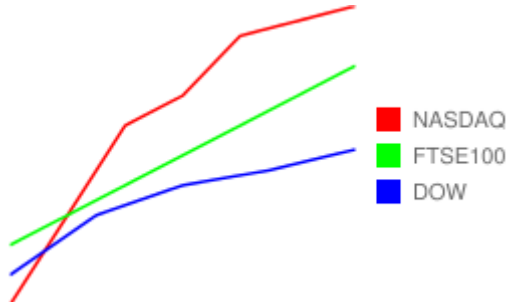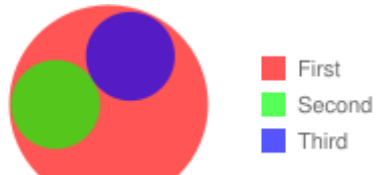
<*color*>

The legend text color, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb).

<*size*>

The point size of the legend text.

Examples

Description	Example
Two examples of legends. Specify legend text in the same order as your data series.	`chdl=NASDAQ|FTSE100|DOW` `chco=FF0000,00FF00,0000FF`

```
chdl=First|Second|Third
chco=ff0000,00ff00,0000ff
```

The first chart demonstrates horizontal legend entries (**chdlp=t**, default layout is horizontal), and the second demonstrates bottom vertical legend entries (**chdlp=bv**).

```
chdl=First|Second|Third
chco=ff0000,00ff00,0000ff
chdlp=t
```

```
chdl=First|Second|Third
chco=ff0000,00ff00,0000ff
chdlp=bv
```

This example demonstrates changing the font size.

```
chdls=0000CC,14
```

Back to top (#top)

Chart Margins chma [*All charts*]

You can specify the size of the chart's margins, in pixels. Margins are calculated inward from the specified chart size (**chs**); increasing the margin size does not increase the total chart size, but rather shrinks the chart area, if necessary.

The margins are by default whatever is left over after the chart size is calculated. This default value varies by chart type. The margins that you specify are a *minimum* value; if the chart area leaves room for margins, the margin size will be whatever is left over; you cannot squeeze the margins smaller than what is required for any legends and labels. Here's a diagram showing the basic parts of a chart:
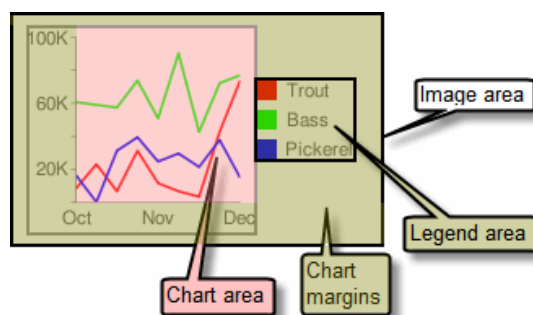
The *chart margins* include the *axis labels* and the *legend area*. The legend area resizes automatically to fit the text exactly, unless you specify a larger width using **chma**, in which case it will expand the margin size wider, squeezing the chart area smaller. You cannot crop a legend by specifying a size that is too small, but you can make it take up more space than it needs.

Tip: In a bar chart, if the bars have a fixed size (the default), the chart area width cannot be reduced. You must specify a smaller or resizeable bar size using chbh (https://developers.google.com/chart/image/docs/gallery/bar_charts?hl=es#chbh).

Syntax

```
chma=
  <left_margin>,<right_margin>,<top_margin>,<bottom_margin>|<opt_legend_width>,<opt_legend_height>
```

<left_margin>, <right_margin>, <top_margin>, <bottom_margin>

Minimum margin size around the chart area, in pixels. Increase this value to include some padding to prevent axis labels from bumping against the borders of the chart.

<opt_legend_width>, <opt_legend_height>

[*Optional*] Width of the margin around the legend, in pixels. Use this to avoid having the legend bump up against the chart area or the edges of the image.
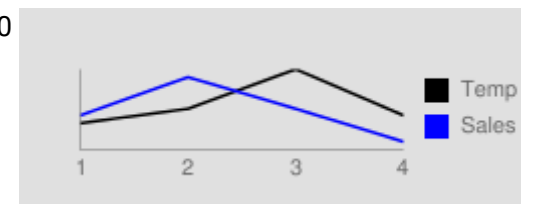
Examples

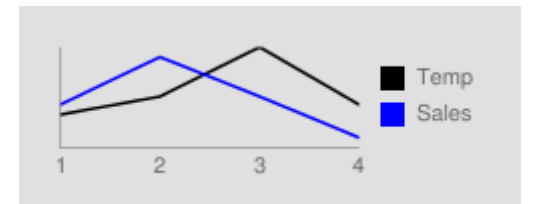| Description	Example
In this example, the chart has a minimum margin of 30 pixels on each side. Because the chart legend is more than 30 pixels wide, the margin on the right side is set to the width of the chart legend, and is different from the other margins. Axis labels are outside the plot area, and are therefore drawn within the margin space.	 `chma=30,30,30,30`
To add a margin around the legend, set a value for the *`<opt_legend_width>`* and *`<opt_legend_height>`* parameters. In this example, the legend is approximately 60 pixels wide. If you set the the *`<opt_legend_width>`* to 80 pixels, the margin extends to 20 pixels outside the legend.	 `chma=20,20,20,30|80,20`

Back to top (#top)

Axis Styles and Labels [*Line, Bar, Google-o-meter, Radar, Scatter*]

You can specify which axes to display on the chart, and give them custom labels and positions, ranges, and styles.

Not all charts show axis lines by default. You can specify exactly which axes your chart should show using the `chxt` parameter. Default axis lines do not show numbers; you must specify an axis in the `chxt` parameter to show numbers.

You can choose to have your axes display numbers reflecting the data values, or you can specify custom axes. The default is to display numeric values, with values scaled to range from 0—100. However, you can change that range using `chxr` to display any range, and you can style the values (for example, to show currency symbols or decimal places) using `chxs`.

If you choose to use custom values, for example: "Mon, Tues, Wed", you can use the `chxl` parameter. To place these labels in specific locations along the axis, use the `chxp` parameter.

Finally, you can use the `chxs` and `chxtc` parameters to specify color, size, alignment, and other properties of both custom and numeric axis labels.

A note on string values: Only URL-safe characters are permitted in label strings. To be safe, you should URL-encode any strings containing characters not in the character set `0-9a-zA-Z`. You can find a URL encoder in the Google Visualization Documentation (https://developers.google.com/chart/interactive/docs/querylanguage?hl=es#plainText).

This section covers the following topics:

- Visible Axes (#axis_type) (`chxt`) - Which axes to display.
- Axis range (#axis_range) (`chxr`) - Value range for each axis.
- Custom Axis Labels (#axis_labels) (`chxl`) - Custom values to show on the axis.
- Axis label positions (#axis_label_positions) (`chxp`) - Placement of custom labels along each axis.
- Axis label styles (#axis_label_styles) (`chxs`) - Color, size, alignment, and formatting of axis labels.
- Axis tick mark styles (#axis_tick_marks) (`chxtc`) - Length of tick marks for a specific axis.

Visible Axes `chxt`

Bar, line, radar, and scatter charts show one or two axis lines by default, but these lines do not include values. To display values on your axis lines, or to change which axes are shown, you must use the `chxt` parameter. By default, the axis values range from 0-100, unless you scale them explicitly using the `chxr` property. To hide all axis lines in a line chart, specify `:nda` after the chart type value in the `cht` parameter (example: cht=`lc:nda`).

By default, the top and bottom axes do not show tick marks by the values, while the left and right axes do show them. You can change this behavior using the chxs (#axis_label_styles) parameter.

Syntax

```
chxt=
  <axis_1>
    ,...,
  <axis_n>
```

<axis>

An axis to show on the chart. Available axes are:

- x - Bottom x-axis
- t - Top x-axis [*Not supported by Google-o-Meter*]
- y - Left y-axis
- r - Right y-axis [*Not supported by Google-o-Meter*]

You can specify multiple axes of the same type, for example: `cht=x,x,y`. This will stack two sets of x-axes along the bottom of the chart. This is useful when adding custom labels along an axis that shows numeric values (see the example below). Axes are drawn from the inside out, so if you have `x,x`, the first x refers to the innermost copy, the next x refers to the next outwards copy, and so on.
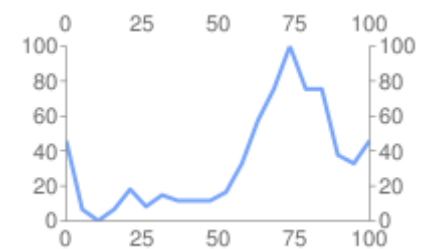
Examples

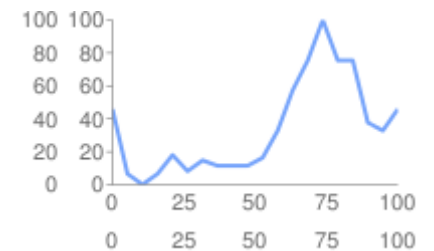| Description	Example
This example shows a line chart with an x-axis, a y-axis, a top axis (t), and a right axis (r). Because no labels are specified, the chart defaults to a range of 0 to 100 for all axes. Note that by default, the top and bottom axes don't show tick marks by the labels.	 `chxt=x,y,r,t`

You can include multiple sets of labels for each axis by including the same value more than once. This example shows two sets of x and two sets of y-axes. This isn't particularly useful when using only the default axis labels, as is shown here. But you can specify custom labels for each copy of each axis, using the `chxl` parameter.	
	`chxt=x,x,y,y`
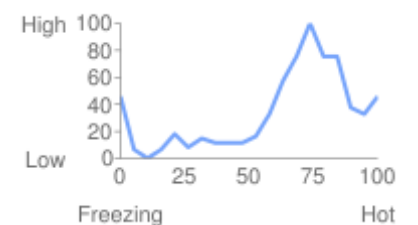	
	`chxt=x,x,y,y`
	`chxl=`
	`  1:|Freezing|Hot|`
	`  3:|Low|High`

This example shows a horizontal bar chart with an x-axis, a y-axis, an upper t-axis, and a right r-axis. Axis labels are omitted, so the Chart API displays a range of 0 to 100 for the x-axis and for the t-axis. The range for the y-axis and for the r-axis is determined by the number of bars. In this case, there are five bars, so the Chart API displays a range of 0 to 4. The first label is centered at the base of the first bar, the second label is centered at the base of the second bar, and so on.	
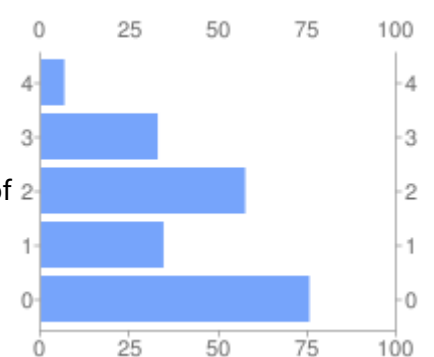
You can suppress default axes in a line chart by specifying `:nda` after the chart type.

```
cht=lc:nda
```

Back to top (#top)

Axis Range `chxr`

You can specify the range of values that appear on each axis independently, using the chxr parameter. Note that this does *not* change the scale of the chart elements, only the scale of the axis labels. If you want to make the axis numbers describe the actual data values, set *<start_val>* and *<end_val>* to the lower and upper values of your data format range, respectively. See Axis Scaling (https://developers.google.com/chart/image/docs/data_formats?hl=es#axis_scale) for more information.

You must make an axis visible using the chxt (#axis_type) parameter if you want to specify its range.

To specify custom axis values, use the chxl (#axis_labels) parameter.

Syntax

Separate multiple axis label ranges using the pipe character (|).

```
chxr=
  <axis_index>,<start_val>,<end_val>,<opt_step>
    |...|
  <axis_index>,<start_val>,<end_val>,<opt_step>
```

<axis_index>

> Which axis to apply the labels to. This is a zero-based index into the axis array specified by `chxt`. For example, the r-axis would be 1 in `chxt=x,r,y`.

<start_val>

> A number, defining the low value for this axis.

<end_val>

> A number, defining the high value for this axis.

<opt_step>

> [*Optional*] The count step between ticks on the axis. There is no default step value; the step is calculated to try to show a set of nicely spaced labels.
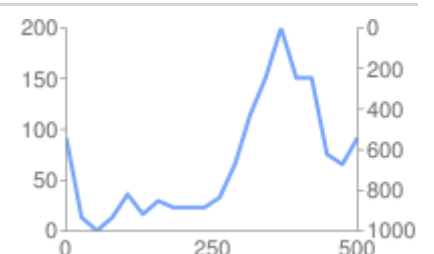
Examples

| Description	Example

This example shows left and right y-axes (**y** and **r**) and one x-axis (**x**).
 	
Each axis has a defined range. Because no labels or positions are specified, values are taken from the given range, and are evenly spaced within that range. In the line chart, values are evenly spread along the x-axis.
 	
Axis direction is reversed for the r-axis (index **2**), because the first value (**1000**) is larger than the last value (**0**).
	
	`chxt=x,y,r`
	`chxr=`
	`  0,0,500|`
	`  1,0,200|`
	`  2,1000,0`

In this example, values are specified for the x-axis.

Axis labels are evenly spaced along the axis. A value of five (**5**) is specified for the *<opt_step>* parameter.


```
chxt=x
chxr=0,10,50,5
```
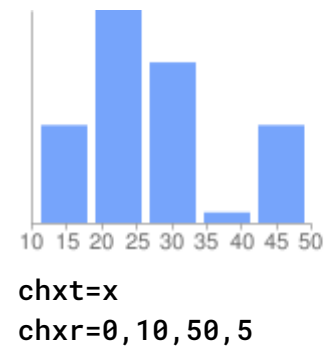
Custom Axis Labels `chxl`

You can specify custom string axis labels on any axis, using the `chxl` parameter. You can specify as many labels as you like. If you display an axis (using the `chxt` parameter) and do not specify custom labels, the standard, numeric labels will be applied. To specify a custom numeric range, use the chxr (#axis_range) parameter (#axis_range) instead.

To set specific locations along the axis for your labels, use the `chxp` parameter.

Syntax

Specify one parameter set for each axis that you want to label. Separate multiple sets of labels using the pipe character (`|`).

```
chxl=
  <axis_index>:|<label_1>|...|<label_n>
    |...|
  <axis_index>:|<label_1>|...|<label_n>
```

<axis_index>

> Which axis to apply labels to. This is an index into the `chxt` parameter array. For example, if you have `chxt=x,x,y,y` then index 0 would be the first x-axis, 1 would be the second x-axis.

<label_1>| ... |<label_n>

> One or more labels to place along this axis. These can be string or number values; strings do not need to be in quotes. *label_1* is displayed at the lowest position on the axis, and *label_n* is displayed at the highest position. Additional labels are spaced evenly between them. Indicate spaces with a + mark. There is no way to specify a line break in a label. Separate labels with a pipe character. **Note:** Do not place a pipe after the final label in the `chxl` parameter.

Examples

| Description	Example

This chart shows how to add custom labels to two axes. Note how the values are evenly spaced, and how the last `chxl` value does not end with a pipe.	
	```chxt=x,y```
	```chxl=```
	```0:|Jan|Feb|March|April|May|```
	```1:|Min|Mid|Max```

This example includes axis labels on the left and right y-axes (**y** and **r**). It also includes two sets of values for the x-axis (**x**). You could consider adding tick marks on the y-axis using chxs (#axis_label_styles).	
	```chxt=x,y,r,x```
	```chxl=```
	```0:|Jan|July|Jan|July|Jan|```

```
1:|0|50|100|
2:|A|B|C|
3:|2005|2006|2007
```

This example includes axis labels on the left and right y-axes (**y** and **r**). It also includes two sets of values for the x-axis (**x**). Note the empty labels for the lower x-axis set, used to space the values apart.

This example uses default values for the axis labels on the left y-axis.

```
chxt=x,y,r,x
chxl=
0:|Jan|July|Jan|July|Jan|
2:|A|B|C|
3:|2005||2006||2007
```

If you want to add a generic label to describe a whole axis (for example, to label one axis "cost" and another "student"), use the **chxt** property to add an additional axis on each side, then use **chxl** to add a single custom label to each side, and **chxp** to space it in the middle of the axis.
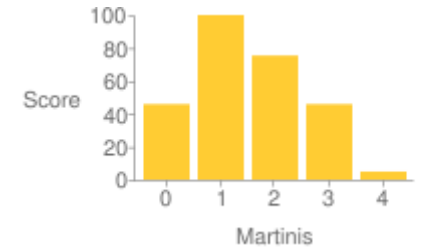
```
chxt=x,x,y,y
chxl=1:|Martinis|3:|Score
chxp=1,50|3,50
```

Back to top (#top)

Axis Label Positions `chxp`

You can specify which axis labels to show, whether using the default labels or custom labels specified using `chxl`. If you do not specify exact positions using this parameter, labels will be spaced evenly and at a default step value along the axes. If you do not specify `chxl`, then the tick mark labels will be the default values (typically data values, or the bar numbers in bar charts).

Syntax

Separate multiple positioning sets using the pipe character (`|`).

```
chxp=
  <axis_1_index>,<label_1_position>,...,<label_n_position>
    |...|
  <axis_m_index>,<label_1_position>,...,<label_n_position>
```

<axis_index>

The axis for which you are specifying positions. This is an index into the `chxt` parameter array. For example, if you have `chxt=x,x,y,y` then index 0 would be the first x-axis, 1 would be the second x-axis, and so on.

<label_1_position>,...,<label_n_position>

The position of the label along the axis. This is a comma-separated list of numeric values, where each value sets the position of the corresponding label in the `chxl` array: the first entry applies to the first label, and so on. The position is a value in the range for that axis (https://developers.google.com/chart/image/docs/data_formats?hl=es#axis_scale). Note that this will always be 0—100 unless you have specified a custom range using `chxr`. You must have as many positions as you have labels for that axis.
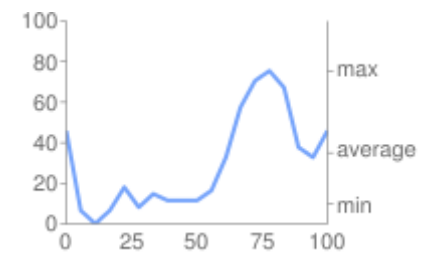
Examples

Description	Example

This example includes r-axis labels at specified positions on the chart. The label text is specified using the `chxl` parameter.

Labels with a specified position of `0` are placed at the bottom of the y- or r-axis, or at the left of the x- or t-axis.

Labels with a specified position of `100` are placed at the top of the y- or r-axis, or at the right of the x- or t-axis.

```
chxt=x,y,r
chxl=2:|min|average|max
chxp=2,10,35,75
```

This example demonstrates showing the default label values, but only at specified locations.

chxp=1,10,35,75 - The y-axis should show only three labels: 10, 35, and 75. Because no custom label text is specified, these axis values are shown. Note how you don't have to space labels evenly apart when you use **chxp**. If **chxp** had not been specified here, the default label value distance on the y-axis would be every 20 units, as shown in the second graph.
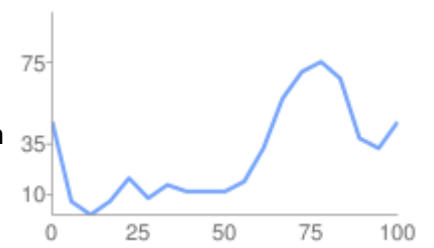

```
chxt=x,y
chxp=1,10,35,75
```
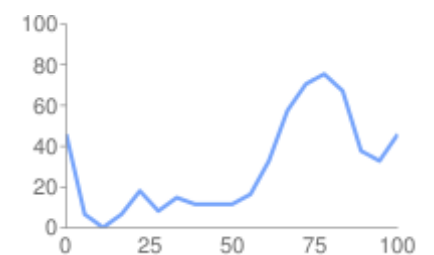


```
chxt=x,y
```
chxp not specified

Back to top (#top)

Axis Label Styles `chxs`

You can specify the font size, color, and alignment for axis labels, both custom labels and default label values. All labels on the same axis have the same format. If you have multiple copies of an axis, you can format each one differently. You can also specify the format of a label string, for example to show currency symbols or trailing zeroes.

By default, the top and bottom axes do not show tick marks by the values, while the left and right axes do show them.

Syntax

Values for multiple axes should be separated using a pipe character (`|`).

```
chxs=
 <axis_index><opt_format_string>,<opt_label_color>,<opt_font_size>,<opt_alignment>,<opt_axis_or_tick>,<opt_tick_colo
   |...|
 <axis_index><opt_format_string>,<opt_label_color>,<opt_font_size>,<opt_alignment>,<opt_axis_or_tick>,<opt_tick_colo
```

<axis_index>

The axis to which this applies. This is a zero-based index into the `chxt` parameter.

<opt_format_string>

[*Optional*] This is an optional format string that, if used, follows immediately after the axis index number without an intervening comma. It starts with a literal letter N followed by the following values, all optional: The formatting string syntax is as follows:
        N*<preceding_text>***<number_type><decimal_places>*zs*<x or y>***<following_text>*
Here is the meaning of each element:

- ***<preceding_text>*** - Literal text to precede each value.

- ***...*** - An optional block wrapped in literal asterisks, in which you can specify formatting details for numbers. The following values are supported, and are all optional:

    - ***<number_type>*** - The number format, for numeric values. Choose one of the following:

        - **f** - [*Default*] Floating point format. Consider specifying precision as well with the *<decimal_places>* value.

- **p** - Percentage format. A % sign is appended automatically. **Note:** When using this format, data values from 0.0 — 1.0 map to 0 — 100% (for example, 0.43 will be shown as 43%).

- **e** - Scientific notation format.

- **c<CUR>** - Format the number in the currency specified, with the appropriate currency marker. Replace **<CUR>** with a three-letter currency code. Example: **cEUR** for Euros. You can find a list of codes on the ISO web site (http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=46121), although not all symbols are supported.

  - **<decimal_places>** - An integer specifying how many decimal places to show. The value is rounded (not truncated) to this length. *Default is 2.*

  - **z** - Display trailing zeros. *Default is no.*

  - **s** - Display group separators. *Default is no.*

  - **x** or **y** -Display the data from the x- or y-coordinate, as specified. The meaning of x data varies by chart type: experiment with your chart to determine what it means. *Default is 'y'.*

- **<following_text>** - Literal text to follow each value.

<opt_label_color>

The color to apply to the axis text (but not axis line), in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). Axis line color is specified separately using *opt_axis_color. Default is gray.*

<opt_font_size>

[*Optional*] specifies the font size in pixels. This parameter is optional.

<opt_alignment>

[*Optional*] Label alignment. For top or bottom axes, this describes how the label aligns to the tick mark above or below it; for left or right axes, this describes how the aligns inside its bounding box, which touches the axis. Specify one of the following numbers:

- **–1** - *Top or bottom*: labels are to the right of the ticks; *Left or right*: labels are left-aligned in their area. *Default for r-axis labels.*

- **0** - *Top or bottom*: labels are centered on the ticks; *Left or right*: labels are centered in their area. *Default for x- and t-axis labels.*

- **1** - *Top or bottom*: labels are to the left of the ticks; *Left or right*: labels are right-aligned in their area. *Default for y-axis labels.*

<opt_axis_or_tick>

[*Optional; not supported in Google-o-meter*] Whether to show tick marks and/or axis lines for this axis. Tick marks and axis lines are only available for innermost axes (for example, they are not supported for the outer of two x-axes). Use one of the following values:

- **l** (lowercase 'L') - Draw axis line only.

- **t** - Draw tick marks only. Tick marks are the little lines next to axis labels.

- **lt** - [*Default*] Draw both an axis line and tick marks for all labels.

- **_** - (Underscore) Draw neither axis line nor tick marks. If you want to hide an axis line, use this value.

<tick_color>

[*Optional; not supported in Google-o-meter*] The tick mark color, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). *Default is gray.*
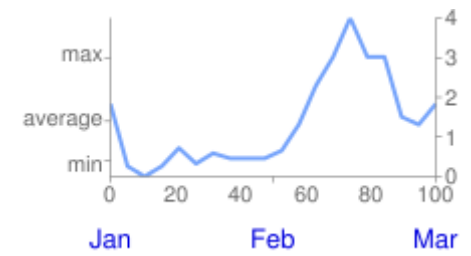
<opt_axis_color>

[*Optional*] The color of this axis line, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). *Default is gray.*

Examples

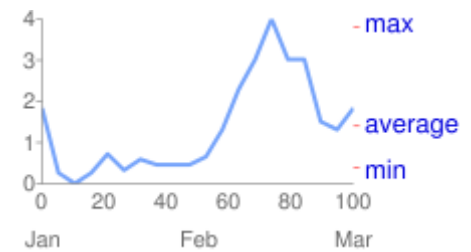| Description	Example
Font size and color are specified for the second x-axis (Jan, Feb, Mar).	

```
chxt=x,y,r,x
chxr=2,0,4
chxl=3:|Jan|Feb|Mar|
     1:|min|average|max
chxp=1,10,35,75
chxs=3,0000DD,13,0,t
```

Font size, color, and alignment are specified for the right y-axis. Tick marks, but no axis line, are drawn.
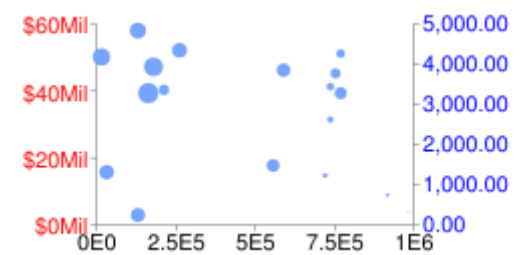


```
chxt=x,y,r,x
chxl=3:|Jan|Feb|Mar|
     2:|min|average|max
chxp=2,10,35,95
chxs=2,0000DD,13,-1,t,FF0000
```

This chart includes three data sets, and shows three sets of axis labels, one per series. Each set of labels is formatted using a custom formatting string, as described here:


```
chd=s:
  984sttvuvkQIBLKNCAIi,
  DEJPgq0uov17zwopQODS,
  AFLPTXaflptx159gsDrn
chxr=
  0,0,1000000,250000|
  1,0,60|
  2,0,5000
chxs=
  0N*e,000000|
  1N*cUSD*Mil,FF0000|
  2N*sz2*,0000FF
```

- `0N*e,000000|`
  - `0` means the first data series
  - `N` means a formatting string
  - `*` means the start of the format specifiers
  - `e` means scientific notation
  - `*` means the end of the format specifiers
  - `000000` means black text.
- `1N*cUSD*Mil,FF0000|`
  - `1` means the second series
  - `N` means a formatting string
  - `*` means the start of the format specifiers
  - `c` means a currency marker
  - `USD` specifies the US dollar as the currency marker to use
  - `*` means the end of the format specifiers
  - `Mil` is a literal following string
  - `FF0000` means red text.
- `2N*sz2*,0000FF`
  - `2` means the third series
  - `N` means a formatting string
  - `*` means the start of the format specifiers
  - `s` means to show grouping specifiers (in US English locale, that is a comma every three zeroes)
  - `z2` means show two trailing zeroes
  - `0000FF` means blue text.

The axis label ranges are set using the `chxr` parameter (*axis_index*, *start*, *end*, *step*). If not set, they would have been 0—100 by default.

[Back to top](#top) (#top)

Axis Tick Mark Styles `chxtc`

You can specify long tick marks for specific axes. Typically this is used to extend a tick mark across the length of a chart. Use the `chxs` parameter to change the tick mark color.

Values for multiple axes should be separated using a pipe character (|). Values within a series should be separated by a comma.

Syntax

```
chxtc=
  <axis_index_1>,<tick_length_1>,...,<tick_length_n>
    |...|
  <axis_index_m>,<tick_length_1>,...,<tick_length_n>
```

<axis_index>

> The axis to which this applies. This is a zero-based index into the `chxt` parameter. Separate values for different axes using a bar delimiter.

<tick_length_1>,...,<tick_length_n>

> Length of the tick marks on that axis, in pixels. If a single value is given, it will apply to all values; if more than one value is given, the axis tick marks will cycle through the list of values for that axis. Positive values are drawn outside the chart area and cropped by the chart borders. The maximum positive value is 25. Negative values are drawn inside the chart area and cropped by the chart area borders.

Examples

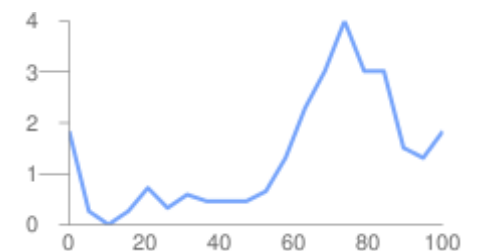| Description	Example

Example of using `chxtc` to create long red tick marks. The tick mark length here exceeds the chart area width, but is cropped to fit within the chart.
 	
 • `chxt=x,y,r,x` - Show a left axis, a right axis, and two bottom axes.
 • `chxl=2:|min|average|max|3:|Jan|Feb|Mar` - The label text assigned to the 'r' (right side) and outer x-axes.
 • `chxp=2,10,35,95` - Custom label positions along the r-axis (index=2) for the three labels.
 • `chxs=2,0000dd,13,-1,t,FF0000` - Axis label styles for the r-axis: text color, text size, left-aligned, with red tick marks.
 • `chxtc=1,10|2,-180` - Axis tick lengths for the y- and r-axes. The first value specifies 10-pixel-long ticks, outside the axis. The second value specifies 180-pixel-long ticks inside the axis; the negative number means that the tick goes inside the axis, and the tick is cropped to fit inside the chart.
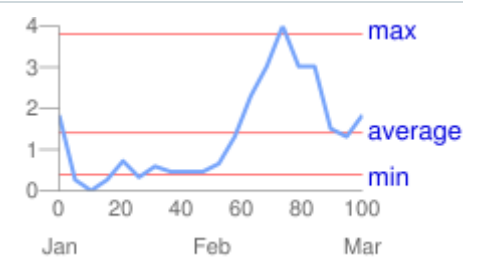	
	 ```chxt=x,y,r,x```
	 ```chxl=```
	 ``` 2:|min|average|max|```
	 ``` 3:|Jan|Feb|Mar```
	 ```chxp=2,10,35,95```
	 ```chxs=```
	 ``` 2,0000dd,13,-1,t,FF0000```
	 ```chxtc=1,10|2,-180```

This chart demonstrates alternating tick lengths. `chxtc` specifies two tick length values for the y-axis (5 and 15), and the ticks drawn on the chart alternate between the two values.	
	 ```chxt=x,y```
	 ```chxtc=```
	 ``` 1,5,15```

Back to top (#top)

Background Fills `chf` [*All charts*]

You can specify fill colors and styles for the chart data area and/or the whole chart background. Fill types include solid fills, striped fills, and gradients. You can specify different fills for different areas (for example, the whole chart area, or just the data area). The chart area fill overwrites the background fill. All fills are specified using the `chf` parameter, and you can mix different fill types (solids, stripes, gradients) in the same chart by separating values with pipe character (|). Chart area fills overwrite chart background fills.

Solid Fills `chf` [*All Charts*]

You can specify a solid fill for the background and/or chart area, or assign a transparency value to the whole chart. You can specify multiple fills using the pipe character (`|`). (Maps: background only).

Syntax

```
chf=<fill_type>,s,<color>|...
```

<fill_type>

> The part of the chart being filled. Specify one of the following values:
>
> - `bg` - Background fill
>
> - `c` - Chart area fill. *Not supported for map charts.*
>
> - `a` - Make the whole chart (including backgrounds) transparent. The first six digits of *<color>* are ignored, and only the last two (the transparency value) are applied to the whole chart and all fills.
>
> - `b<index>` - Bar solid fills (bar charts only). Replace *<index>* with the series index of the bars to fill with a solid color. The effect is similar to specifying `chco` in a bar chart. See Bar Chart Series Colors (https://developers.google.com/chart/image/docs/gallery/bar_charts?hl=es) for an example.
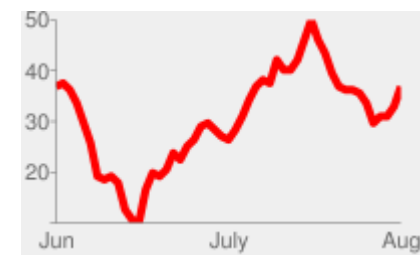
s

> Indicates a solid or transparency fill.

<color>

> The fill color, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). For transparencies, the first six digits are ignored, but must be included anyway.
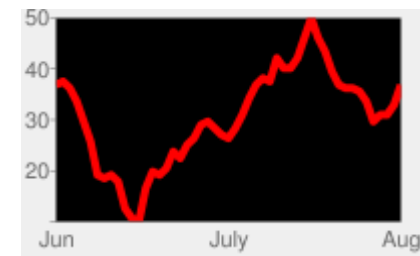
Examples

Description	Example
This example fills the chart background with pale gray (**EFEFEF**).	![chart with pale gray background] `chf=bg,s,EFEFEF`

This example fills the chart background with pale gray (**EFEFEF**) and fills the chart area in black (**000000**).	![chart with black chart area]
	
	`chf=c,s,000000|`
	`bg,s,EFEFEF`

This example applies a 50% transparency to the whole chart (80 in hexadecimal is 128, or about 50% transparency). Notice the table cell background showing through the chart.	![chart with transparency] `chf=a,s,00000080`

Back to top (#top)

Gradient Fills `chf` [*Line, Bar, Google-o-meter, Radar, Scatter,Venn*]

You can apply one or more gradient fills to chart areas or backgrounds. Gradient fills are fades from a one color to another color. (Pie, Google-o-meter charts: background only.)

Each gradient fill specifies an angle, and then two or more colors anchored to a specified location. The color varies as it moves from one anchor to another. You must have at least two colors with different *<color_centerpoint>* values, so that one can fade into the other. Each additional gradient is specified by a *<color>,<color_centerpoint>* pair.

Syntax

```
chf=<fill_type>,lg,<angle>,<color_1>,<color_centerpoint_1>
  ,...,
  <color_n>,<color_centerpoint_n>
```

<fill_type>

>   The chart area to fill. One of the following:
>   - `bg` - Background fill
>   - `c` - Chart area fill.
>   - `b<index>` - Bar gradient fills (bar charts only). Replace *<index>* with the series index of the bars to fill with a gradient. See Bar Chart Series Colors (https://developers.google.com/chart/image/docs/gallery/bar_charts?hl=es) for an example.

lg

>   Specifies a gradient fill.

<angle>

>   A number specifying the angle of the gradient from 0 (horizontal) to 90 (vertical).

<color>

>   The color of the fill, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb).

<color_centerpoint>

>   Specifies the anchor point for the color. The color will start to fade from this point as it approaches another anchor. The value range is from 0.0 (bottom or left edge) to 1.0 (top or right edge), tilted at the angle specified by *<angle>*.
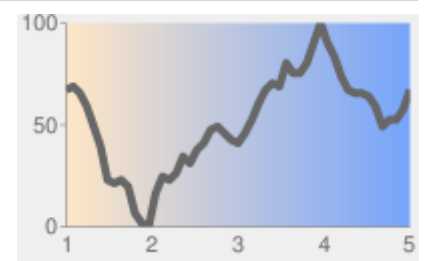
Examples

| Description	Example

Chart area has a horizontal linear gradient, specified with an angle of zero degrees (`0`).
 	
The colors are peach (`FFE7C6`), centered on the left side (position `0.0`) and blue (`76A4FB`) centered on the right side (position `1.0`).
 	
The chart background is drawn in gray (`EFEFEF`).
	
	`chf=`
	  `c,lg,0,`
	  `FFE7C6,0,` (peach)
	  `76A4FB,1` (blue)

Chart area has a diagonal (bottom left to top right) linear gradient, specified with an angle of forty-five degrees (`45`).
 	
Peach (`FFE7C6`) is the first color specified. The bottom left of the chart is pure peach.
 	
Blue (`6A4FB`) is the second color specified. The top right of the chart is pure blue. Note how we specify an offset of 0.75, to provide a peak of blue that fades away towards the top right corner.
 	
The chart background is drawn in gray (`EFEFEF`).
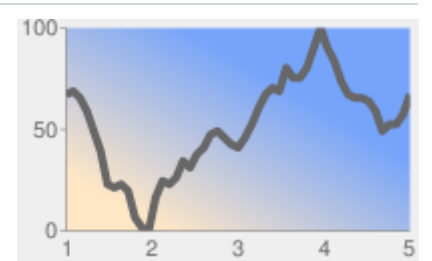	
	`chf=`
	  `c,lg,45,`

```
                                          FFE7C6,0,  (peach)
                                          76A4FB,0.75 (blue)
```
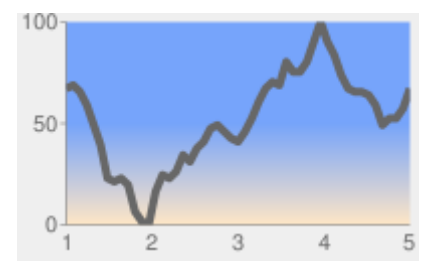
Chart area has a vertical (top to bottom) linear gradient, specified with an angle of ninety degrees (`90`).

Blue (`76A4FB`) is the first color specified. The top of the chart is pure blue.

Peach (`FFE7C6`) is the second color specified. The bottom of the chart is pure peach.

The chart background is drawn in gray (`EFEFEF`).


```
chf=
  c,lg,90,
  FFE7C6,0,  (peach)
  76A4FB,0.5 (blue)
```

Back to top (#top)

Striped fills `chf` [*Line, Bar, Google-o-meter, Radar, Scatter, Venn*]

You can specify a striped background fill for your chart area, or the whole chart. (Pie, Google-o-meter charts: background only.)

Syntax

```
chf=                                                              ○●
  <fill_type>,ls,<angle>,<color_1>,<width_1>
    ,...,
  <color_n>,<width_n>
```

<fill_type>

> The chart area to fill. One of the following:
> - `bg` - Background fill
> - `c` - Chart area fill
> - `b`*<index>* - Bar striped fills (bar charts only). Replace *<index>* with the series index of the bars to fill with stripes. See Bar Chart Series Colors (https://developers.google.com/chart/image/docs/gallery/bar_charts?hl=es) for an example.

ls

> Specifies linear stripe fill.

<angle>

> The angle of all stripes, relative to the y-axis. Use `0` for vertical stripes or `90` for horizontal stripes.

<color>

> The color for this stripe, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). Repeat *<color>* and *<width>* for each additional stripe. **You must have at least two stripes.** Stripes alternate until the chart is filled.

<width>

> The width of this stripe, from `0` to `1`, where `1` is the full width of the chart. Stripes are repeated until the chart is filled. Repeat *<color>* and *<width>* for each additional stripe. **You must have at least two stripes.** Stripes alternate until the chart is filled.

Examples

Description	Example
- `bg,ls,0` - Background stripe fill with stripes at a zero degree angle to the y-axis (parallel to the y-axis). The stripes fill the chart background as well as the plot area. - `CCCCCC,0.15` - The first stripe is dark gray, 15% as wide as the chart. - `FFFFFF,0.1` - The second stripe is white, 10% as wide as the chart.	
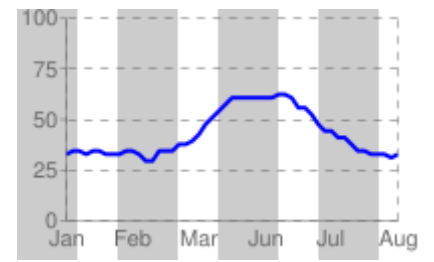
```
chf=
    bg,ls,0,
    CCCCCC,0.15,
    FFFFFF,0.1
```

- **c,ls,90** - Chart area with horizontal stripes at an angle of ninety degrees from the y-axis. The stripes fill the plot area, but the chart background is omitted.
- **999999,0.25** - The first stripe is dark gray, 25% as wide as the chart.
- **CCCCCC,0.25** - Same as the first stripe, but a lighter gray.
- **FFFFFF,0.25** - Same as the first stripe, but white.
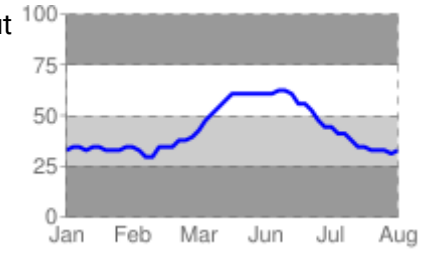

```
chf=
    c,ls,90,
    999999,0.25,
    CCCCCC,0.25,
    FFFFFF,0.25
```

Back to top (#top)

Grid Lines `chg` [*Line, Bar, Radar, Scatter*]

You can specify solid or dotted grid lines on your chart using the `chg` parameter.

This parameter doesn't let you specify the thickness or color of the lines. For more ways to make lines across your chart, see shape markers (#gcharts_shape_markers) (`chm` type h, H, v, or V), range markers (#gcharts_range_markers) (`chm`), and axis tick marks (#axis_tick_marks) (`chxtc`).

Syntax

```
chg=
    <x_axis_step_size>,<y_axis_step_size>,<opt_dash_length>,<opt_space_length>,<opt_x_offset>,<opt_y_offset>
```

<x_axis_step_size>, <y_axis_step_size>

> Used to calculate how many x or y grid lines to show on the chart. 100 / step_size = how many grid lines on the chart. So: 20,25 would mean 5 vertical grid lines and 4 horizontal grid lines.

<opt_dash_length>, <opt_space_length>

> [*Optional*] Used to define dashed grid lines. The first parameter is the length of each line dash, in pixels. The second parameter is the spacing between dashes, in pixels. Specify 0 for *<opt_space_length>* for a solid line. *Default values are 4,1.*
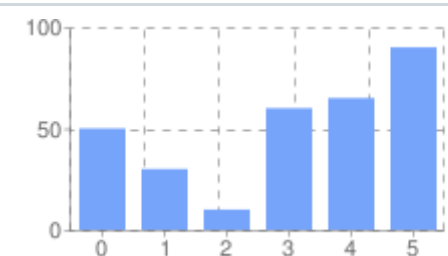
<opt_x_offset>,<opt_y_offset>

> [*Optional*] The number of units, according to the chart scale, to offset the x and y grid lines, respectively. Can be positive or negative values. If you specify this value, you must also specify all preceding values. *Default values are 0,0.*
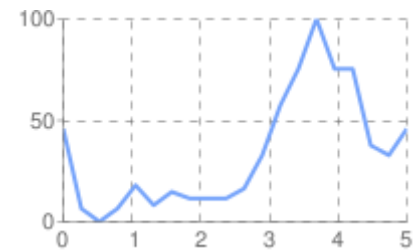
Examples

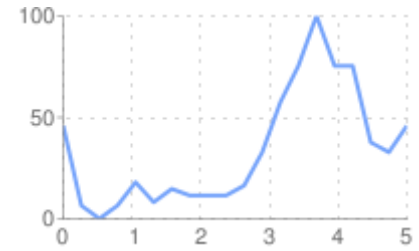| Description	Example
These examples use only the *<x_axis_step_size>* and *<y_axis_step_size>* parameters. The Chart API displays a dashed grid line by default.	

chg=20,50

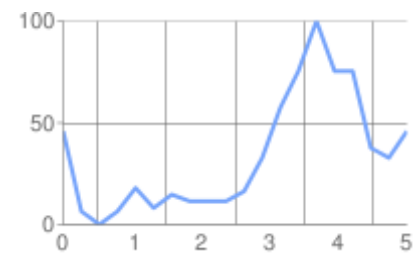This example uses larger spaces to display lighter grid lines (1,5).

chg=20,50,1,5

To display solid grid lines, specify zero (0) for the **<opt_space_length>** parameter.
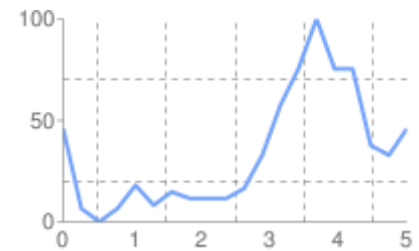
This chart also specifies an x-axis offset of 10.

chg=20,50,1,0,10

This chart demonstrates an x-axis offset of 10, and a y axis offset of 20.

chg=20,50,3,3,10,20

Back to top (#top)

Line Styles `chls` [*Line, Radar*]

You can specify line thickness and solid/dashed style with the `chls` parameter. This parameter can only be used to style lines in line or radar charts; you cannot use it to style the line in a compound chart lines, unless the base type of the compound chart is a line chart.

Syntax

Separate multiple line styles by the pipe character (|); the first style applies to the first line, the second to the next, and so on. If you have fewer styles than lines, the default style is applied to all the unspecified lines.

```
chls=
  <line_1_thickness>,<opt_dash_length>,<opt_space_length>
    |...|
  <line_n_thickness>,<opt_dash_length>,<opt_space_length>
```

<line_1_thickness>

> Thickness of the line, in pixels.

<opt_dash_length>, <opt_space_length>

> [*Optional*] Used to define dashed grid lines. The first parameter is the length of each line dash, in pixels. The second parameter is the spacing between dashes, in pixels. For a solid line, specify neither value. If you only specify <opt_dash_length>, then <opt_space_length> will be set to <opt_dash_length>. *Default is 1,0 (a solid line)*.

Examples

Description	Example

Here the dashed line is specified by **3,6,3** and the thicker, solid line is specified by **5**..

<div style="text-align:right">
chls=3,6,3|5
</div>
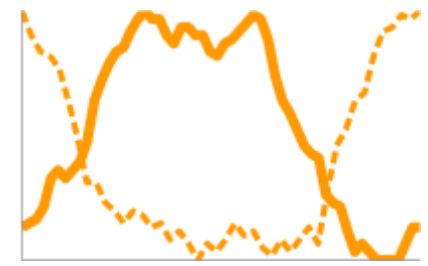
Dynamic Icon Markers `chem` [*Bar, Line, Radar, Scatter*]

Create your chart, and specify one or more dynamic icons as chem values. The syntax of `chem` is as follows. All semicolon-delimited items in purple are optional, and any one can be omitted entirely in your URL. You can include multiple markers by including multiple syntax strings delimited by a | character. You can read more about dynamic icons on the <u>dynamic icon page</u> (https://developers.google.com/chart/image/docs/gallery/dynamic_icons?hl=es).

You can also embed a chart inside another chart as a dynamic icon. See the <u>Embedded Charts</u> (#embedded_charts) subsection below.

```
chem=
  y;s=<icon_string_constant>;d=<marker_data_string>;ds=<which_series>;dp=<which_points>;py=<opt_z_order>;po=<x,y>;of=
  |...|
  y;s=<icon_string_constant>;d=<marker_data_string>;ds=<which_series>;dp=<which_points>;py=<opt_z_order>;po=<x,y>;of=
```

s=*<icon_string_constant>*

A string marker constant for a dynamic icon, from the <u>dynamic icon page</u> (https://developers.google.com/chart/image/docs/gallery/dynamic_icons?hl=es). This constant is *almost* the same as the `chst` parameter for freestanding icons. But where the freestanding icon string starts with "d_", you should remove that prefix to get the equivalent dynamic icon marker. **Example:** freestanding icon: `d_bubble_icon_text_small`; equivalent dynamic icon marker: `bubble_icon_text_small`.

d=*<marker_data_string>*

The data required for this particular marker type. This holds the same string that would be used in a `chld` parameter for an equivalent freestanding icon, *except* that all | delimiters should be replaced by commas (remember to use commas instead of pipe markers for multiline text!). Note that within the data string, you must also escape the following characters with a @ mark: pipe (|), at (@), equals (=), comma (,), semicolon (;). Examples: `hello@,+world`, `5@@10+cents+each`.

ds=*<which_series>*

[*Optional*] The zero-based index of the data series that this marker belongs to. *Default value is 0.*

dp=*<which_points>*

[*Optional*] Specifies which data points are used to draw markers. *Default value is 0 (first point in the series)*. Use one of the following formats:

- $n.d$ - Which data point to draw the marker on, where $n.d$ is the zero-based index in the series. If you specify a non-integer value, then the fraction indicates a calculated intermediate point. For example, 3.5 means halfway between point 3 and point 4.
- `range,<start>,<end>,<step>` - Draw a marker on every *step* data point in a range from *start* to *end*, inclusive. start and end are index values, and can be floating point numbers to indicate intermediate values. All values are optional; defaults are: start=0, end=last item, step=1. If you skip a value, you must still include any intermediate commas, but you don't have to include empty trailing commas. Examples: `dp=range,0,4` draws a marker on elements 0 through 4; `dp=range,5,10,2` draws a marker on elements 5, 7, and 9; `dp=range,2` draws points on the third point and later; `dp=range,3,,1.5` draws markers on every 1.5 data points from the fourth item to the last.
- `all` - Draw a marker on every element. This is equivalent to `range,0,`*end_index*. Example: `dp=all`
- `every,n` - Draw a marker on every n[th] marker. Example: `dp=every,2` draws a marker on items 0, 2, and 4.

py=*<z_order>*

[*Optional*] The layer on which to draw the marker, compared to other markers and all other chart elements. This is a floating point number from -1.0 to 1.0, inclusive, where -1.0 is the bottom and 1.0 is the top. Chart elements (lines and bars) are just below zero. If two markers have the same value, they are drawn in the order given by the URL. *Default value is 0.0* (just above the chart elements).

po=<*x,y*>

[*Optional*] An absolute position on the chart at which to draw the marker. x and y are two floating point numbers, where 0.0,0.0 is the bottom left corner and 1.0,1.0 is the top right corner.

of=<*x_offset,y_offset*>

[*Optional*] A number of pixels to offset the icon from its normal position. *x_offset* and *y_offset* are positive or negative integers. It is important to specify this value in an embedded dynamic icon, because the marker will be centered vertically and horizontally over the point, which means that the point probably won't line up with the data marker. A good offset for an upright pin is `of=0,22`; a good offset for a slanted pin is either `of=-12,20` or `of=12,20` depending on the direction of the slant, but you might have to experiment. *Default value is 0,0.*

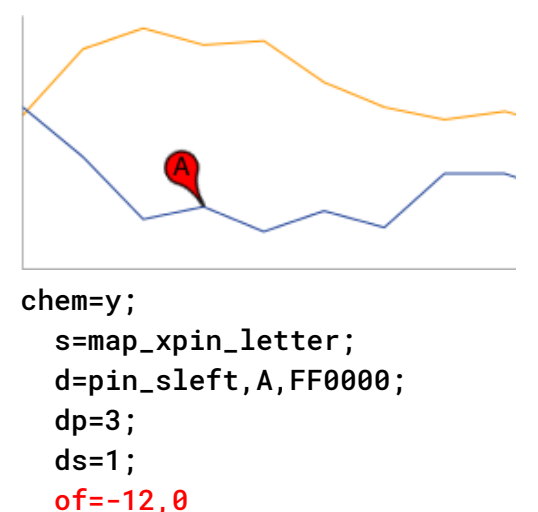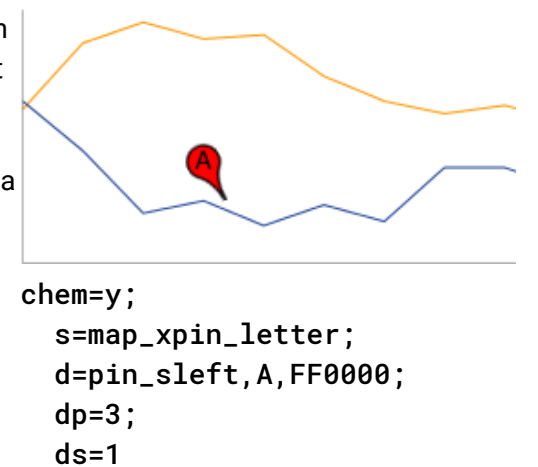## Examples

| Description	Example

Here are examples of the same dynamic icon created as a free-standing image, and used as a marker in a line chart.
 	
Chart 1: `https://chart.googleapis.com/chart?chs=300x140&cht=lc&chco=FF9900,224499&`
`chd=t:75,74,66,30,10,5,3,1&chls=1|1&`
`chem=y;s=bubble_icon_text_small;d=ski,bb,Wheeee!,FFFFFF;dp=2;ds=0&chm=v,ccccFF,0,::.2,2`
 	
Chart 2: `https://chart.googleapis.com/chart?`
`chst=d_bubble_icon_text_small&chld=ski|bb|Wheeee!|FFFFFF|000000`
	
	`chem=`
	    `y;s=bubble_icon_text_small;c`
	
	`chst=`
	    `d_bubble_icon_text_small`
	`chld=`
	    `ski|bb|Wheeee!|FFFFFF|000000`
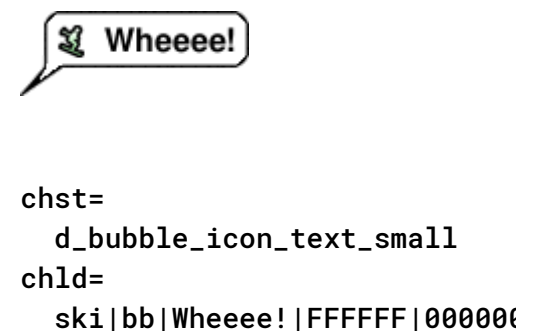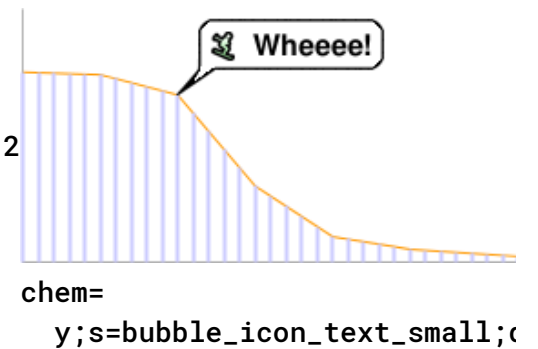
Note that a dynamic icon marker will be centered on the point horizontally and vertically. So if you use a dynamic icon with a tail, the tail won't be on the chart point; the marker center will be. The first chart centers the mark on data point 3, which actually puts the tail a bit to the right of the mark on the line.
 	
The second chart uses the `of` value to move the tail of the marker on top of the desired data point. A good offset for a slanted pin is -12,20.
	
	`chem=y;`
	    `s=map_xpin_letter;`
	    `d=pin_sleft,A,FF0000;`
	    `dp=3;`
	    `ds=1`
	
	`chem=y;`
	    `s=map_xpin_letter;`
	    `d=pin_sleft,A,FF0000;`
	    `dp=3;`
	    `ds=1;`
	    `of=-12,0`

To include multiple dynamic icons, repeat the syntax string, delimited by a | character.

This example shows a range, as well as two individual markers.

Note how the vertical pins are offset by of 0,22, and the slanted pin has an offset of 12,20 to make the pin points line up with the series that they describe.
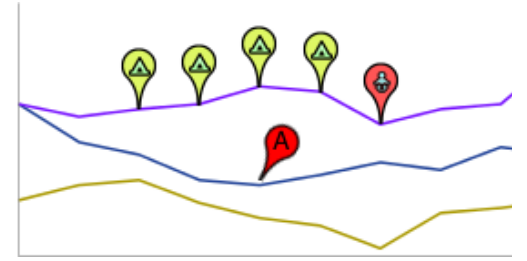

```
chem=
  y;s=map_xpin_letter;d=pin_sr
FF0000;dp=4;ds=0;py=1;of=12,26
  y;s=map_pin_icon;d=baby,FF55
  y;s=map_pin_icon;d=camping,[
```

This demonstrates a multiline text marker. The marker must be offset after adding the text, because the bubble resizes to fit the text, causing it to be re-centered on the chart. Note how newlines in the text are indicated by commas in the **d** data string.
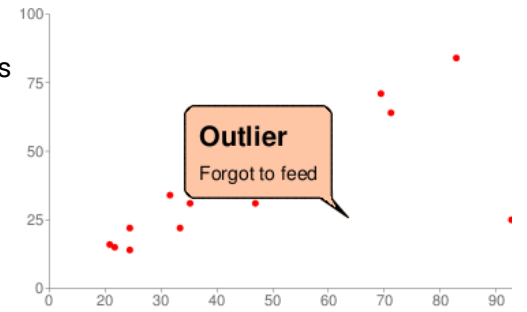

```
chem=y;
  s=bubble_texts_big;
  d=bbbr,FFC6A5,000000,Outlier
  ds=0;
  dp=13;
  of=-120,2
```

Embedded Charts

You can embed one chart inside another using the dynamic icon syntax.

There are two styles of embedded chart markers: embedded charts in a bubble, and embedded charts with no bubble. Here are examples of both:
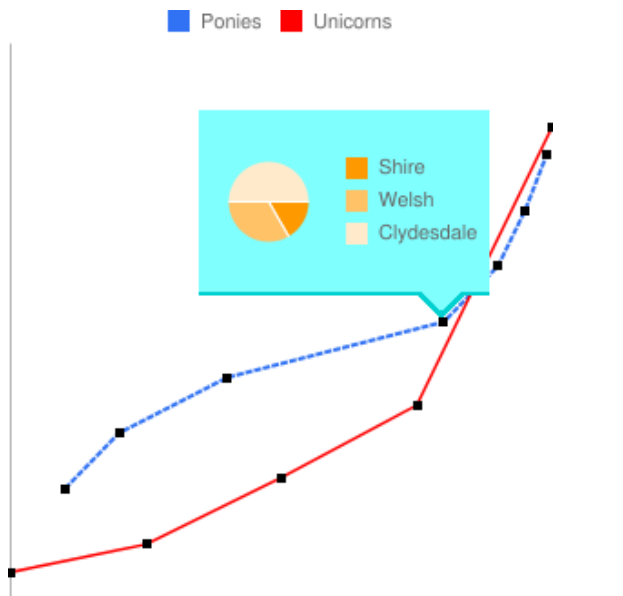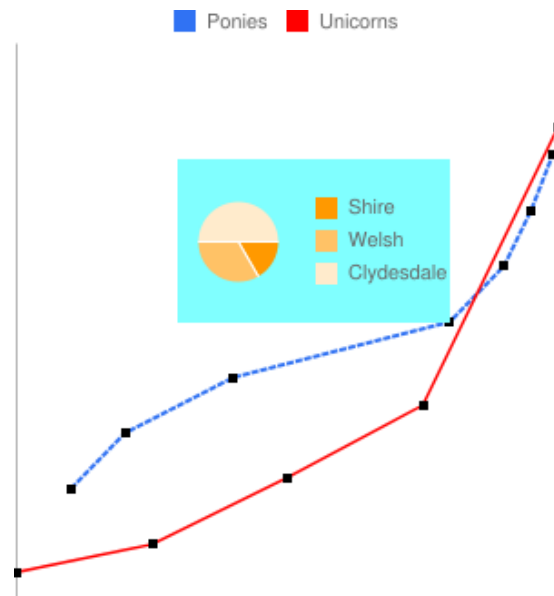

Chart with bubble

Chart without bubble

Here are the specifics of the s=<*icon_string_constant*>;d=<*marker_data_string*> parameters, both for non-bubble and bubble-embedded charts (parameters covered above aren't described again here):
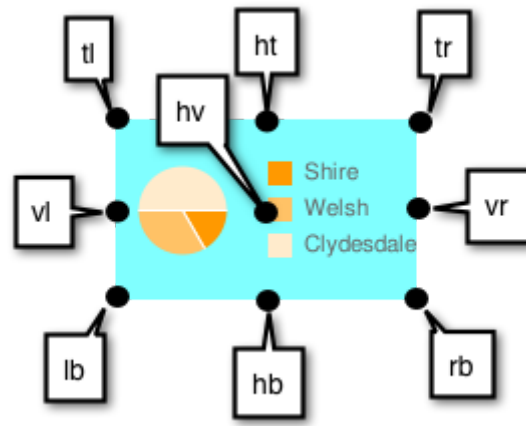
Syntax

```
Non-bubble:
  chem=y;s=ec;d=<alignment_string>,<chart_data>;ds=<which_series>;dp=<which_points>;py=<z_order>;po=<x,y>;of=<x_offse
```

```
Bubble:
  chem=y;s=ecb;d=<frame_type>,<padding>,<frame_color>,<fill_color>,<chart_data>;ds=<which_series>;dp=<which_points>;p
```

alignment_string

[*Non-bubble only*] Which part of the bubble-less embedded chart is pinned to the data point. Choose one of the two-letter string constants in the following diagram:

chart_data

The data for the embedded chart. This is everything after `https://chart.googleapis.com/chart?` in the URL of the chart to embed. Use the tool below, or follow the rules listed below the tool.

frame_type

[*Bubble only*] One of the dynamic icon <u>frame style constants</u> (https://developers.google.com/chart/image/docs/gallery/dynamic_icons?hl=es#frame_style_constants).

padding

[*Bubble only*] Padding inside the bubble, in pixels.

frame_color

[*Bubble only*] Color of the frame, as a six-digit HTML color string without the # mark. Example: `FF00FF`.

fill_color

[*Bubble only*] Bubble fill color, as a six-digit HTML color string without the # mark. Example: `FF00FF`.

Embedded chart data

Use the following conversion tool to help generate your chart string, or else generate the chart string manually following the rules given after the tool.

Chart String
(everything after https://chart.googleapis.com/chart?)

Encode

cht=p&chd=s:Uf9a&chs=75x50

ENCODE >

Rules for manual conversion

1. First replace all the following characters in the parameter and value pairs with the following values, in the order shown:

Replace	With this
%7C or %7c	|
@	@@
%	%25
,	@,
|	@l

;	@;
&	%26
=	%3D

2. Then replace all the `&` and `=` values in the `parameter1=value1&parameter2=value2...` pairs with commas.

Line Fills **chm** [*Line, Radar*]

You can fill the area below a data line with a solid color.

You can combine line fills with any other **chm** parameters using a pipe character (|) to separate the **chm** parameters.

Syntax

```
chm=
  <b_or_B>,<color>,<start_line_index>,<end_line_index>,<0>
    |...|
  <b_or_B>,<color>,<start_line_index>,<end_line_index>,<0>
```

<b_or_B>

Whether to fill to the bottom of the chart, or just to the next lower line.

- B - Fill from *<start_line_index>* to the bottom of the chart. *<end_line_index>* supports a special syntax to let you fill a segment of the chart. This is easiest if you have a chart with a single line that you want to fill.

- b - Fill between two lines in a multi-line chart. Start and end lines are indicated by *<start_line_index>* and *<end_line_index>*.

<color>

An RRGGBB format hexadecimal number (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb) of the fill color.

<start_line_index>

The index of the line at which the fill starts. The first data series specified in **chd** has an index of zero (**0**), the second data series has an index of **1**, and so on.

<end_line_index>

- Fill type 'b' - The line at which to stop the fill. This line must be below the current line.

- Fill type 'B' - One of the following choices:

  - *any value* - Any single number in this parameter is ignored, and the fill will go from the specified line to the base of the chart

  - *start:end* - To fill a vertical slice below the chart, specify *start:end*, where these are data point indices describing where to start and stop the fill. Both values are optional, and default to *first_point:last_point*. (See example below.)

<0>

Reserved — must be zero.

Examples

Description	Example
For a single series, it is simplest to use **chm=B**. This fills the entire area under the line.	
This is the only fill area type available for radar charts. **chm=b** does not work with radar charts.	

```
chm=B,76A4FB,0,0,0
```

Here's a chart with two lines and two fills. Filling the area below the bottom line and above the top line requires a special technique, covered next.
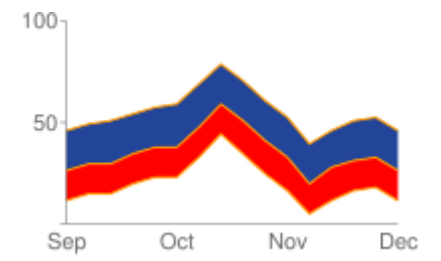

```
chd=s:
  cefhjkqwrlgYcfgc,
  QSSVXXdkfZUMRTUQ,
  HJJMOOUbVPKDHKLH,

chm=
  b,224499,0,1,0| (blue)
  b,FF0000,1,2,0| (red)
```

In a multi-line chart, to fill from the top of the chart to the first line, include a series that contains two copies of the highest data value for the data format. So, for example, "`....`" for extended encoding, "`100,100`" for basic text format, and so on. Fill from this series to the topmost line.

To fill from the last line to the bottom of the chart, include a series that contains two copies of the lowest data value for the data format. So, for example, AAAA for extended encoding, 0,0 for basic text format, and so on.

The lines themselves are drawn in black, using `chco`.
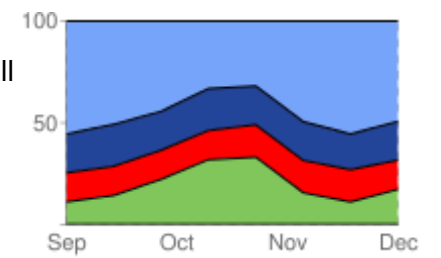

```
chd=e:
  ...., (highest value)
  cefhjkqwrlgYcfgc,
  QSSVXXdkfZUMRTUQ,
  HJJMOOUbVPKDHKLH,
  AAAA (zero value)
chm=
  b,76A4FB,0,1,0| (light
blue)
  b,224499,1,2,0| (blue)
  b,FF0000,2,3,0| (red)
  b,80C65A,3,4,0 (green)
chco=000000
```
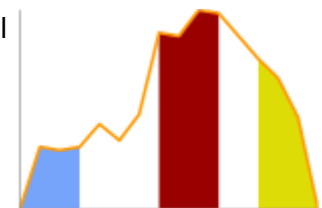
Here is a line chart with vertical fill slices. This is accomplished by specifying a *start:end* pair for *<end_line_index>* with line fill type 'B'.
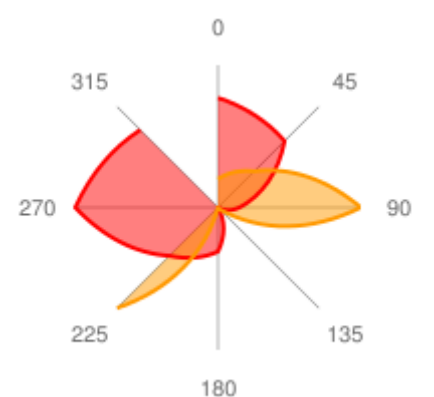
- `B,76A4FB,0,0:3,0` - Blue vertical fill under line 0, bound by data points 0—3.

- `B,990000,0,7:10,0` - Red vertical fill under line 0, bound by data points 7—10.

- `B,DEDC06,0,12:,0` - Yellow vertical fill under line 0, from data point 12 through the end of the series.


```
chm=
  B,76A4FB,0,0:3,0|
  B,990000,0,7:10,0|
  B,DEDC06,0,12:,0
```

This example shows using a line fill on a radar chart.


```
chm=
  B,FF000080,0,1.0,5.0|
  B,FF990080,1,1.0,5.0
```

Shape Markers `chm` [*Bar, Line, Radar, Scatter*]

You can specify graphical markers for all or individual data points on a chart. If two or more markers occupy the same point, the markers are drawn in the order in which they appear in the `chm` parameter. You can also create text markers on data points, which is covered in Data Point Markers (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_data_point_labels).

You can combine shape markers with any other `chm` parameters using a pipe character (|) to separate the `chm` parameters.

Syntax

Specify one set of the following parameters for each series that should be marked. To mark multiple series, create additional parameter sets, delimited by a pipe character. You do not need to mark up all series. If you do not assign markers to a data series, it will not get any markers.

Shape markers behave slightly differently in scatter charts (https://developers.google.com/chart/image/docs/gallery/scatter_charts?hl=es#chart_types). See that documentation for more information.

```
chm=
  [@]<marker_type>,<color>,<series_index>,<opt_which_points>,<size>,<opt_z_order>,<opt_offset>
    |...|
  [@]<marker_type>,<color>,<series_index>,<opt_which_points>,<size>,<opt_z_order>,<opt_offset>
```

@

> [*Optional*] If you precede the marker type with the optional @ character, then *<opt_which_points>* should use the *x:y* format.

<marker_type>

> The type of marker to use. Specify one of the following types:
>
> - `a` - Arrow
> - `c` - Cross
> - `C` - Rectangle. If a rectangle marker, you must have at least two data series, where series 0 specifies the bottom edge and series 1 specifies the top edge. *<size>* specifies the width of the rectangle, in pixels.
> - `d` - Diamond
> - `E` - Error-bar marker ($\mathbb{I}$) This marker requires two data series to create, one value for the bottom, and the corresponding point in the second series for the top. It also exposes an extended *<size>* syntax: *line_thickness*[:*top_and_bottom_width*] where *top_and_bottom_width* is optional. See the examples below.
> - `h` - Horizontal line across the chart at a specified height. (**The only valid format for *<opt_which_points>* parameter is *n.d.*.**)
> - `H` - Horizontal line through the specified data marker. This supports an extended *<size>* syntax that lets you specify an exact line length: *line_thickness*[:*length*] where :*length* is optional, and defaults to the full chart area width.
> - `o` - Circle
> - `s` - Square
> - `v` - Vertical line from the x-axis to the data point
> - `V` - Vertical line of adjustable length. This supports an extended *<size>* value syntax that lets you specify an exact line length: *line_thickness*[:*length*] where :*length* is optional, and defaults to the full chart area height. The marker is centered on the data point.
> - `x` - An X

<color>

> The color of the markers for this series, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb).

<series_index>

> The zero-based index of the data series on which to draw the markers. Ignored for `h` markers and markers that specify location by x/y position (start with the @ character). You can use hidden data series as a source for markers; see Compound Charts (https://developers.google.com/chart/image/docs/gallery/compound_charts?hl=es) for more information. Grouped vertical bar charts support a special extended syntax (https://developers.google.com/chart/image/docs/gallery/bar_charts?hl=es#markerplacementtarget) to align markers with specific bars.

<opt_which_points>

[*Optional*] Which point(s) to draw markers on. *Default is all markers*. Use one of the following values:

- `n.d` - Where to draw the marker. The meaning depends on the marker type:

  - *All types except h* - Which data point to draw the marker on, where *n.d* is the zero-based index in the series. If you specify a non-integer value, then the fraction indicates a calculated intermediate point. For example, 3.5 means halfway between point 3 and point 4.

  - h - A number from 0.0 to 1.0, where 0.0 is the bottom of the chart, and 1.0 is the top of the chart.

- `-1` - Draw a marker on all data points. You can also leave this parameter empty to draw on all data points.

- `-n` - Draw a marker on every *n*-th data point. Floating point value; if n is less than 1 the chart will calculate additional intermediary points for you. For example, -0.5 will put twice as many markers as data points.

- `start:end:n` - Draw a marker on every *n*-th data point in a range, from *start* to *end* index values, inclusive. All parameters are optional (may be absent), so 3::1 would be from the fourth element to the last, step 1, and omitting this parameter entirely would default to *first:last*:1. All values can be floating point numbers. *start* and *end* can be negative, to count backward from the last value. If both *start* and *end* are negative, be sure that they are listed in increasing value (for example, -6:-1:1). If the *n* step value is less than 1, it will calculate additional data points by interpolating the data values given. *Default values are first:last:1*

- `x:y` - Draw a marker at a specific x/y point on the chart. This point does not have to be on a line. Add the @ character before the marker type to use this option. Specify the coordinates as floating point values, where `0:0` is the bottom left corner of the chart and `1:1` is the top right corner of the chart. For example, to add a red, 15-pixel diamond to the center of a chart, use `@d,FF0000,0,0.5:0.5,15`.

<size>

The size of the marker, in pixels. Most take a single number value for this parameter; the V, H, and S markers support the syntax *<size>*[:*width*] where the optional second part specifies the line or marker length.

<opt_z_order>

[*Optional*] The layer on which to draw the marker, compared to other markers and all other chart elements. This is a floating point number from -1.0 to 1.0, inclusive, where -1.0 is the bottom and 1.0 is the top. Chart elements (lines and bars) are just lower than zero. If two markers have the same value, they are drawn in the order given by the URL. *Default value is 0.0* (just above the chart elements).

<opt_offset>

[*Optional*] Let you specify horizontal and vertical offsets from the specified location. Here is the syntax, which uses a : delimiter: `reserved:<horizontal_offset>:<vertical_offset>`. If specified, you can include an empty „ value in the `chm` parameter string for *<opt_z_order>*. Examples: o,FF9900,0,4,12,,`:10` o,FF9900,0,4,12.0,,`:-10:20` o,FF9900,0,4,12,1,`::20`

- *reserved* - Leave blank.

- `<horizontal_offset>` - A positive or negative number specifying the horizontal offset, in pixels. Optional; leave blank if not used.

- `<vertical_offset>` - A positive or negative number specifying the vertical offset, in pixels. Optional; leave blank if not used.

Examples

| Description	Example

Here's an example of several of the shape and line markers.
 	
• `a,990066,0,0.0,9.0` - Purple arrow, first series, first point, size 9.
• `c,FF0000,0,1.0,20.0` - Red cross, first series, second point, size 9.
• `d,80C65A,0,2,20.0` - Green diamond, first series, third point, size 9.
• `H,000000,0,3,1:40` - Black horizontal line, first series, data point 3, one pixel wide, forty pixels long.
• `o,FF9900,0,4.0,12.0` - Orange circle, first series, fifth point, size 12.
• `s,3399CC,0,5.0,11.0` - Blue square, first series, sixth point, size 11.
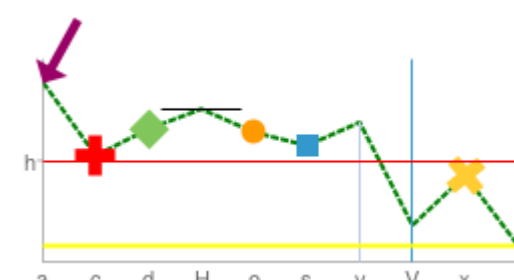	
	chm=
	  a,990066,0,0.0,9.0|
	  c,FF0000,0,1.0,20|

- `v,BBCCED,0,6.0,1.0` - Vertical line up to point, first series, seventh point, one pixel wide.
- `V,3399CC,0,7.0,1.0` - Vertical line bottom to top of chart, first series, eighth point, one pixel wide.
- `x,FFCC33,0,8.0,20.0` - Yellow 'X', first series, ninth point, size 20.
- `H,FFFF00,0,9,2` - Horizontal yellow line the width of the chart at data point 9.
- `h,FF0000,0,0.5,1` - Red horizontal line at designated height, first series, halfway up the chart, one pixel wide.

```
d,80C65A,0,2.0,20.0|
H,000000,0,3,1:40|
o,FF9900,0,4.0,12.0|
s,3399CC,0,5.0,11.0|
v,BBCCED,0,6,1.0|
V,3399CC,0,7,1.0|
x,FFCC33,0,8,20|
H,FFFF00,0,9,2|
h,FF0000,0,0.5,1
```


Here's an example using diamonds for one data series, and circles for the other data series.

If two or more markers occupy the same point, the markers are drawn in the order in which they appear in the **chm** parameter. Here, the circle is the first marker specified with **chm**, so it is drawn first. The diamond is specified and drawn second, which results in it being drawn on top of the circle.
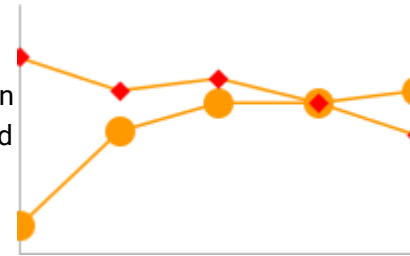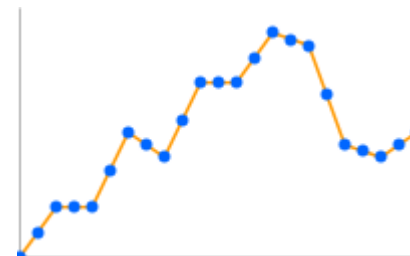


```
chm=
  o,FF9900,0,-1,15.0|
  d,FF0000,1,-1,10.0
```

Here's a line chart with a marker on every second data point (-2 means every other point).
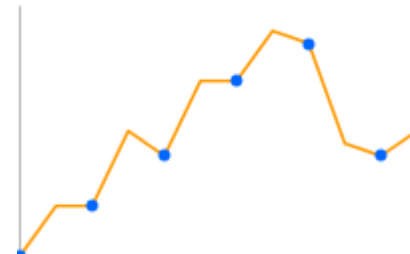

```
chd=t:
  0,20,20,50,40,70,70,90,85,45,40,50
chm=
  o,0066FF,0,-2,6
```

Here's a line chart with twice as many markers as data points (-0.5 means every half point).


```
chd=t:
  0,20,20,50,40,70,70,90,85,45,40,50
chm=
  o,0066FF,0,-.5,6
```


This example shows how to use **h** and **v** markers to create grid lines with custom colors and thickness. The z-order value (the last value) is set to -1 so that the grid lines are drawn beneath the data line.


```
chm=
  h,76A4FB,0,0:1:.2,2,-1|
  V,76A4FB,0,::2,0.5,-1
```


This chart adds vertical fill lines to a line chart:

- **v** - Vertical lines to the chart
- **FF0000** - Red lines
- **0** - Series index
- **: :.5** - Range specifier: from start to end, every 0.5 points.
- **2** - Thickness 2 pixels.
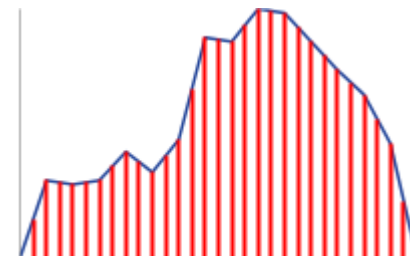


```
chm=
  v,FF0000,0,::.5,2
```

This example adds an arrow and text marker to the chart using exact coordinates. The first D marker is the trace line under the bars. The second marker is the arrow, and the third marker is the arrow text.
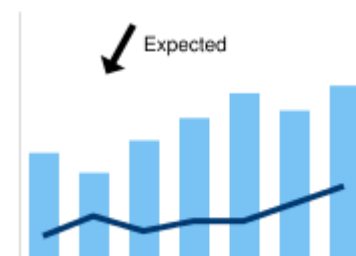

```
chm=
```

```
D,003971,1,0,3|
@a,000000,0,.25:.75,7|
@tExpected,000000,0,.35:.85,10
```

A horizontal line fixed to a specific data point (**H**) can be useful for showing relative values, or emphasizing the height of a data value on a chart.
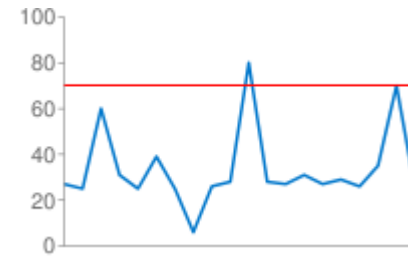

```
chm=H,FF0000,0,18,1
```

This graph demonstrates the markers that can specify line thickness and length in the *<size>* parameter.

- `E,000000,0,6,1:20` - Black error bar with 1 pixel wide lines, top and bottom bars 20 pixels long. The bottom is anchored to series 0 point 8, the top is anchored to series 1 point 8.
- `H,990066,1,2,5:50` - Purple, horizontal line five pixels wide, fifty pixels long centered on data point 2.
- `V,3399CC,0,8,3:50`- Blue, vertical line 3 pixels wide, fifty pixels long, centered on data point 8.
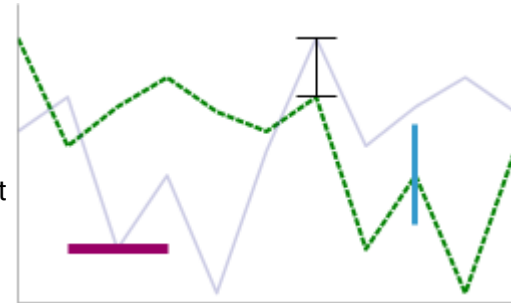

```
chm=
  E,000000,0,6,1:20|
  H,990066,1,2,5:50|
  V,3399CC,0,8,3:50
```

Text and Data Value Markers `chm` [*Bar, Line, Radar, Scatter*]

You can label specific points on your chart with custom text, or with formatted versions of the data at that point.

You can combine any `chm` markers using a pipe character (`|`) to separate the `chm` parameter sets.

A note on string values: Only URL-safe characters are permitted in label strings. To be safe, you should URL-encode any strings containing characters not in the character set `0-9a-zA-Z`. You can find a URL encoder in the Google Visualization Documentation (https://developers.google.com/chart/interactive/docs/querylanguage?hl=es#plainText).

Syntax

Specify one set of the following parameters for each series that should be marked. To mark multiple series, create additional parameter sets, delimited by a pipe character. You do not need to mark up all series. If you do not assign markers to a data series, it will not get any markers.

```
chm=
  <marker_type>,<color>,<series_index>,<opt_which_points>,<size>,<opt_z_order>,<opt_placement>
    |...|
  <marker_type>,<color>,<series_index>,<opt_which_points>,<size>,<opt_z_order>,<opt_placement>
```

<marker_type>

> The type of marker to use. You can choose from the following types:
>
> - `f<text>` - A flag containing text. Specify the character 'f', followed by custom URL-encoded text.
>
>   ⭐ To escape commas in text markers, precede the comma by a \ mark.
>
>   Example: `fHello\,+World!`
>
> - `t<text>` - A simple text marker. Specify the character 't' followed by custom URL-encoded text.
>
>   ⭐ To escape commas in text markers, precede the comma by a \ mark.
>
>   Example: `tHello\,+World!`

- `A<text>` - An annotation marker. This is similar to a flag marker, but markers will coordinate their position so that they do not overlap. The only valid format for *<opt_which_points>* is *n.d*, to signify the index of a point in the series.
- `N<formatting_string>` - The value of the data at this point, with optional formatting. If you do not use the <u>chds</u> (https://developers.google.com/chart/image/docs/data_formats?hl=es#data_scaling) parameter (custom scaling) it gives the exact encoded value; if you do use that parameter *with any format type* the value will be scaled to the range that you specify. See an example of `chds` with numeric markers below. With this marker type in a <u>stacked bar chart</u> (https://developers.google.com/chart/image/docs/gallery/bar_charts?hl=es), if you specify -1 for *<series_index>* you will get a marker that shows the sum of all values in this stacked bar. The formatting string syntax is as follows:

  `<preceding_text>*<number_type><decimal_places>zs<x or y>*<following_text>`.

  All of these elements are optional. Here is the meaning of each element:
  - *<preceding_text>* - Text to precede each value.
  - `*...*` - An optional block wrapped in literal asterisks, in which you can specify formatting details for numbers. The following values are supported, and are all optional:
    - *<number_type>* - The number format, for numeric values. Choose one of the following:
      - `f` - [*Default*] Floating point format. Consider specifying precision as well with the *<decimal_places>* value.
      - `p` - Percentage format. A % sign is appended automatically. **Note:** When using this format, data values from $0.0 - 1.0$ map to $0 - 100\%$ (for example, 0.43 will be shown as 43%).
      - `e` - Scientific notation format.
      - `c<CUR>` - Format the number in the currency specified, with the appropriate currency marker. Replace *<CUR>* with a three-letter currency code. Example: `cEUR` for Euros. You can find a list of codes on the <u>ISO web site</u> (http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=46121), although not all symbols are supported.
    - `<decimal_places>` - An integer specifying how many decimal places to show. The value is rounded (not truncated) to this length. *Default is 2.*
    - `z` - Display trailing zeros. *Default is no.*
    - `s` - Display group separators. *Default is no.*
    - `x` or `y` -Display the data from the x- or y-coordinate, as specified. The meaning of x data varies by chart type: experiment with your chart to determine what it means. *Default is 'y'.*
  - *<following_text>* - Text to follow each value.

<color>

The color of the markers for this set, in <u>RRGGBB hexadecimal format</u> (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb).

<series_index>

The zero-based index of the data series on which to draw the markers. If this is a <u>stacked bar chart</u> (https://developers.google.com/chart/image/docs/gallery/bar_charts?hl=es) and the marker type is N (data point values), you can specify -1 to create a summed value marker for each stack of bars. See below for an example.

<opt_which_points>

[*Optional*] Which point(s) to draw markers on. *Default is all markers.* Use one of the following values:
- *n.d* - Which data point to draw the marker on, where *n.d* is the zero-based index in the series. If you specify a non-integer value, then the fraction indicates a calculated intermediate point. For example, 3.5 means halfway between point 3 and point 4.
- `-1` - Draw a marker on all data points. You can also leave this parameter empty to draw on all data points.
- `-n` - Draw a marker on every *n*-th data point.
- *start:end:n* - Draw a marker on every *n*-th data point in a range, from *start* to *end* index values, inclusive. All parameters are optional (may be absent), so 3::1 would be from the fourth element to the last, step 1, and omitting this parameter entirely would default to *first:last:*1. All values can be floating point numbers. *start* and *end* can be negative, to count backward from the last value. If both *start* and *end* are negative, be sure that they are listed in increasing value (for example, -6:-1:1). If the *n* step value is less than 1, it will calculate additional data points by interpolating the data values given. *Default values are first:last:1*
- *x:y* - [*Not supported for N-type markers*] Draw a marker at a specific x/y point on the chart. This point does not have to be on a line. Add the at character (@) before the marker type to use this option. Specify the coordinates as floating point values, where

`0:0` is the bottom left corner of the chart, `0.5:0.5` is the center of the chart, and `1:1` is the top right corner of the chart. For example, to add a red, 15-pixel diamond to the center of a chart, use `@d,FF0000,0,0.5:0.5,15`.

<size>

The size of the marker in pixels. If this is a scatter chart with a third data series (used to specify point sizes), this value will be scaled by the data range. So if the data range is 0—100 and *<size>* is 30, a data value of 100 would be 30 pixels wide, a data value of 50 would be 15 pixels wide, and so on.

<opt_z_order>

[*Optional*] The layer on which to draw the marker, compared to other markers and all other chart elements. This is a floating point number from -1.0 to 1.0, inclusive, where -1.0 is the bottom and 1.0 is the top. Chart elements (lines and bars) are just lower than zero. If two markers have the same value, they are drawn in the order given by the URL. Default value is 0.0 (just above the chart elements).

<opt_placement>

[*Optional*] Additional placement details describing where to put this marker, in relation to the data point. You can specify horizontal and/or vertical relative positioning, as well as offsets. Placement syntax is a string with : delimiters as shown here. All elements are optional: *<horizontal_and_vertical_justification>:<horizontal_offset>:<vertical_offset>*. If specified, you can include an empty „ value in the `chm` parameter string for *<opt_z_order>*. Examples: `N,000000,0,1,10,,b` and `N,000000,0,1,10,,lv` and `N,000000,0,1,10,,r::10`.

horizontal_and_vertical_justification

The anchor point of the marker. This behaves opposite to justification, so a left anchor actually puts the marker to the *right* of the data point. You can choose a horizontal and/or vertical justifier from the following list:

- **Horizontal placement**: 'l', 'h', or 'r' - Left, center, or right-anchored, horizontally. *Default is 'l'*.
- **Vertical placement**: 'b', 'v', 't' - Bottom, middle, or top-anchored, vertically. *Default is 'b'*.
- **Bar-relative placement [*Bar charts only*]**: 's', 'c', 'e' - Base, center, or top of a bar. For stacked charts, this is relative to the section of the bar for each series, not for the whole bar. If the series index given is -1 (stack total) it is in relation to the whole bar. This can be combined with vertical placement values: for example, 'be' or 'vs'. *Default value is 'e'*.

horizontal_offset

A horizontal offset for this marker, in pixels. *Default is 0*.

vertical_offset

A vertical offset for this marker, in pixels. *Non-bar chart default: 15; bar chart chart default: 2*.


Description	Example

Here's an example of value labels on a bar chart.

The first chart (`N*p0*`) specifies a data value marker, with values shown as a percentage, rounded to zero decimal places, in black, on all values, in 11-point text. Note that the data values are all between 0.0 and 1.0, which, in percentage format, are moved up two decimal places.

The second chart (`N*cEUR1*`) shows the same values formatted as Euro values, one decimal place, in black, on all values, in 11-point text.
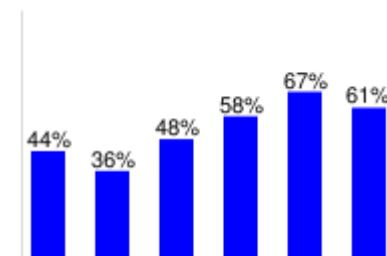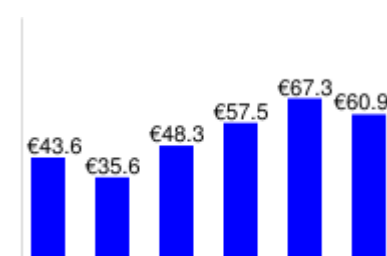


```
chm=
  N*p0*,000000,0,-1,11
chd=t:
  0.4356,0.3562,0.4834,0.575,0.673,0.6091
```



```
chm=
  N*cEUR1*,000000,0,-1,11
```

You can use the **chds** parameter to scale the numeric value displayed. You can use **chds** for all data formats, including simple and extended encoding. This will not affect the bar size or the axis labels, but only the data marker value.

This example shows a chart with simple-encoded values of 46, 39, 29, 30, 43, 41. Simple encoding range is 0−61 . The **chds** parameter defines a custom marker scale of 0−1, so marker values are scaled to this range, but the bar heights are not affected (if this were text format data (https://developers.google.com/chart/image/docs/data_formats?hl=es#data_scaling), the bars would be scaled as well).
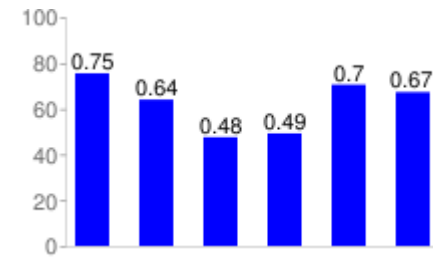


```
chd=s:underp
chm=N,000000,0,-1,11
chds=0,1
```

Here's an example of a chart with a text label at the minimum point and a flag label at the maximum point.
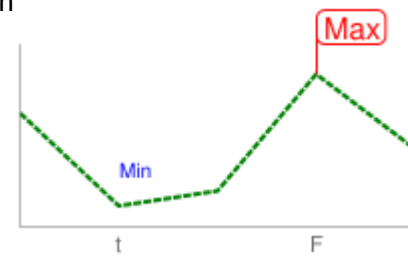


```
chm=
  tMin,0000FF,0,1,10|
  fMax,FF0000,0,3,15
```

This example shows a stacked chart with values for individual series, plus the series total. To show the stacked series values, we must use the 'c' positioning option; if we did not, the top bar value would overlap the sum value at the top of each bar.
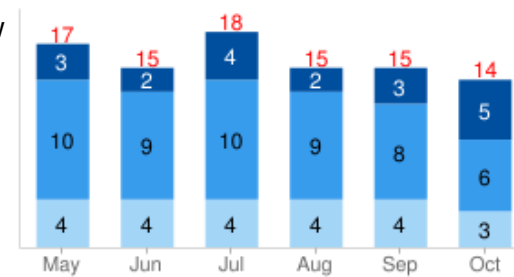


```
chm=
  N,FF0000,-1,,12|
  N,000000,0,,12,,c|
  N,000000,1,,12,,c|
  N,ffffff,2,,12,,c
```

Some more demonstrations of how vertical and horizontal placement work. This example demonstrates various combinations of anchor values for bar charts (which use s, c,and e for vertical placement). Note how a right anchor moves a marker left, and a top anchor moves a marker down, and vice-versa. The red dots show the base, center, and top of each bar. The number is the data value, fixed using different anchor values for each bar.
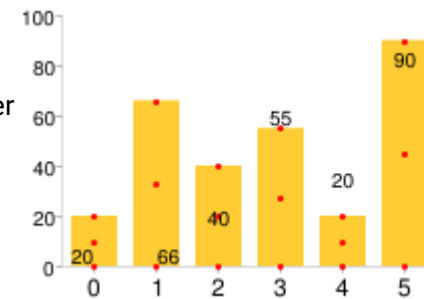


```
chm=
  N,000000,0,0,10,,rs
  N,000000,0,1,10,,ls
  N,000000,0,2,10,,c
  N,000000,0,3,10,,e
  N,000000,0,4,10,,e::15
  N,000000,0,5,10,,e::-12
```

Annotation markers automatically adjust the label position so that they don't overlap. The first **chm** value is for the line fill, the following values are all annotation markers.
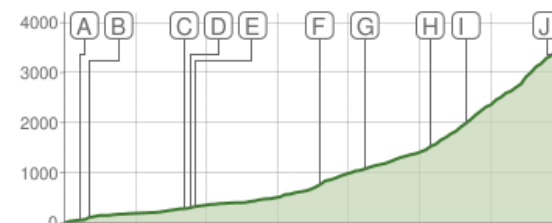

```
chm=B,C5D4B5BB,0,0,0
  AA,666666,0,3,15
  AB,666666,0,5,15
  AC,666666,0,24,15
  AD,666666,0,25,15
  AE,666666,0,26,15
  AF,666666,0,51,15
  AG,666666,0,60,15
  AH,666666,0,73,15
  AI,666666,0,80,15
  AJ,666666,0,99,15
```

Another annotation marker example demonstrating city altitudes in Switzerland.
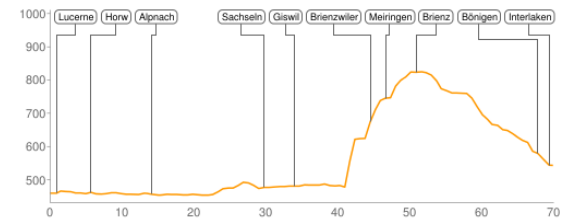
Range Markers `chm` [*Bar, Candlestick, Line, Radar, Scatter*]

You can color horizontal or vertical bands of background fill to highlight specific areas of a chart.

You can combine any `chm` markers using a pipe character (`|`) to separate the `chm` parameter sets.

Syntax

Specify one set of the following parameters for each band to draw. To draw multiple bands, create additional parameter sets, delimited by a pipe character. Ranges are drawn in the order specified, so the last range drawn will be drawn on top of previous ranges.

```
chm=
  <direction>,<color>,0,<start_point>,<end_point>
    |...|
  <direction>,<color>,0,<start_point>,<end_point>
```

<direction>

> Specifies horizontal or vertical shading. Use `r` for a horizontal range and `R` for a vertical range.

<color>

> The range color as an RRGGBB format hexadecimal number
> (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb).

0

> *Reserved — must be zero.*

<start_point>

> The start position of the range.

- For **horizontal** range markers, this is a position on the y-axis, where `0.00` is the bottom of the chart, and `1.00` is the top of the chart.
- For **vertical** range markers, this is a position on the x-axis, where `0.00` is the left of the chart, and `1.00` is the right of the chart.

<end_point>

> The end position of the range.

- For **horizontal** range markers, this is a position on the y-axis, where `0.00` is the bottom of the chart, and `1.00` is the top of the chart.
- For **vertical** range markers, this is a position on the x-axis, where `0.00` is the left of the chart, and `1.00` is the right of the chart.
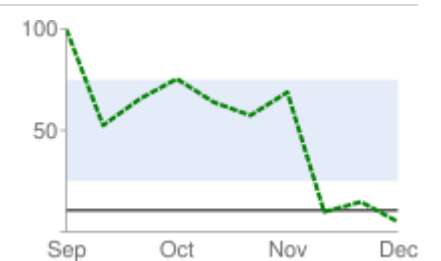
Examples

| Description	Example
Range markers can be a thin line or a band of color. • `r,E5ECF9,0,0.75,0.25` - Range marker, pale blue, (reserved), 0.5 of the height of the chart. • `r,000000,0,0.1,0.11` - Range marker, black, (reserved), starts at 0.1 of the way up the y-axis and ends at 0.11 of the way up the y-axis (a thin black line).	 chm=

```
r,E5ECF9,0,0.75,0.25|
r,000000,0,0.1,0.11
```

This example shows the vertical range markers. The first marker is a red line (**FF0000**), and the second is a pale blue band (**A0BAE9**).
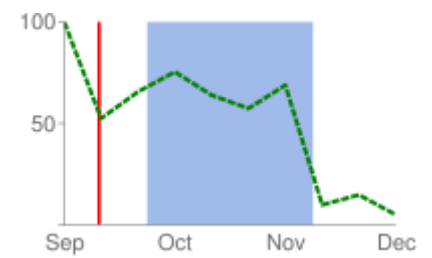


```
chm=
  R,FF0000,0,0.1,0.11|
  R,A0BAE9,0,0.75,0.25
```

Markers are drawn in the order specified. In this example, you can see that the vertical red marker was drawn before the pale blue horizontal marker.
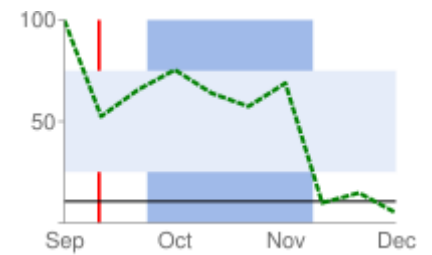

```
chm=
  R,FF0000,0,0.1,0.11|
  R,A0BAE9,0,0.75,0.25|
  r,E5ECF9,0,0.75,0.25|
  r,000000,0,0.1,0.11
```

Here's an example of a line chart that uses range markers to draw faint horizontal lines across the chart at the zero line, midpoint line, and top.
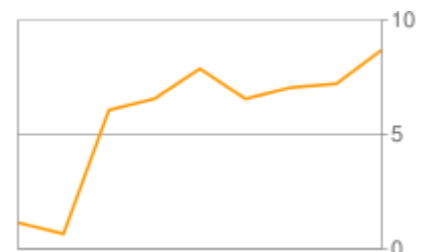

```
chm=
  r,000000,0,0.499,0.
501|
  r,000000,0,0.998,1.0|
  r,000000,0,0.0,0.002
```

Back to top (#top)

Candlestick Markers `chm=F` [*Bar, Line*]

Candlestick markers indicate variance and direction change in a data series. Often they are used to show stock values during the course of a day. The marker includes segments that show the high and low value, as well as the opening and closing value for a specific time period (typically a day). For more about candlestick markers, see underline (http://en.wikipedia.org/wiki/Candlestick_chart).

A candlestick marker is drawn as a rectangle bisected by a vertical line. It requires four data series to draw a candlestick marker; here is what each series specifies:

- **Series 1 and 4** specify the bottom and top of the vertical line, respectively. These typically represent the low and high values for the day.
- **Series 2 and 3** specify the vertical borders of the rectangle. Series 2 is the opening value, and series 3 is the closing value. The color of the rectangle depends on which is higher: when the opening value (series 2) is lower than the closing value (series 3), the price has increased, and the rectangle is filled with solid green by default; when the opening value (series 2) is higher than the closing value (series 3), the price has decreased, and the rectangle is filled with solid red by default. You can only specify a fill color for the rectangle with decreasing value. When you specify that, the rectangle with increasing value is unfilled (empty). Note that series 2 can be either the top or the bottom of the rectangle, depending on whether the price has gone up or down.

You can combine candlestick markers with any other `chm` parameters using a pipe character (`|`) to separate the `chm` parameters.

Note: If you do not want the lines for the data used to draw the markers to appear in the chart, you must include a 0 after the format type. For example: `chd=t0:10,20,30,40` in a text format data string. See Compound Charts (https://developers.google.com/chart/image/docs/gallery/compound_charts?hl=es) for more information.

Here's an example, showing the lines for each series:
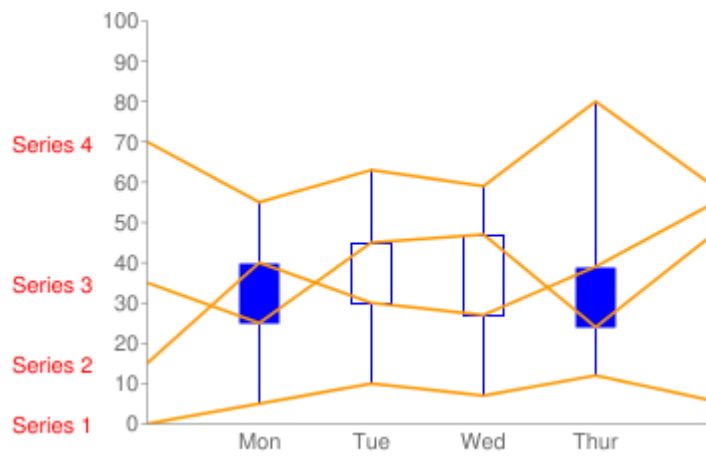
Syntax

```
chm=
  F,<opt_declining_color>,<data_series_index>,<opt_which_points>,<width>,<opt_z_order>
```

F

> Indicates that this is a candlestick marker.

<opt_declining_color>

> [*Optional*] Fill color for the rectangles when the value is decreasing (when series 2 value > corresponding series 3 value). This is an RRGGBB format hexadecimal number (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb). When the values increase, the rectangle will be empty. *Default is solid green for increasing, solid red for decreasing (you cannot specify a custom fill color for increasing values)*.

<data_series_index>

> The index of the data series to use as the first series for your candlestick markers. This is a zero-based index. So, if you specify 1 here, and you have six series, the second, third, fourth and fifth will be used to draw candlestick markers.

<opt_which_points>

> [*Optional*] Specifies which data points are used to draw markers. *Default is all markers*. Use one of the following formats:
> - `n.d` - Draw a marker on a single point in the series, where *n.d* is the index of the point in the series. If you specify a non-integer value, then the fraction indicates a calculated intermediate point. For example, 3.5 means halfway between point 3 and point 4.
> - `-1` - Draw a marker on all data points. You can also leave this parameter empty to draw on all markers.
> - `-n` - Draw a marker on every *n*-th data point.
> - `start:end:n` - Draw a marker on every *n*-th data point in a range, from *start* to *end* index values, inclusive. All parameters are optional (may be absent), so 3::1 would be from the fourth element to the last, step 1, and omitting this parameter entirely would default to *first:last*:1. All values can be floating point numbers. *start* and *end* can be negative, to count backward from the last value. If both *start* and *end* are negative, be sure that they are listed in increasing value (for example, -6:-1:1). If the *n* step value is less than 1, it will calculate additional data points by interpolating the data values given. *Default values are first:last:1*

<width>

> The width of all rectangles, in pixels.

<opt_z_order>

> [*Optional*] The layer on which to draw the marker, compared to other markers and all other chart elements. This is a floating point number from -1.0 to 1.0, inclusive, where -1.0 is the bottom and 1.0 is the top. Chart elements (lines and bars) are just lower than zero. If two markers have the same value, they are drawn in the order given by the URL. *Default value is 0.0* (just above the chart elements).
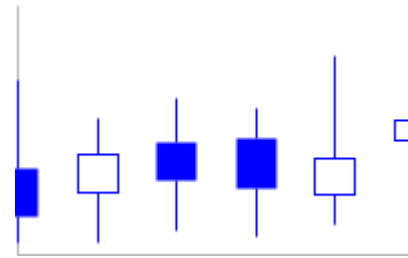
Examples

Description	Example

Here's an example of candlestick markers on a line chart with four series. The custom fill color `0000FF`

(blue) is specified, and this color is used to fill the rectangles when the point in series 3 is smaller than the equivalent point in series 2.

The first and last rectangle are trimmed by the chart. To eliminate these values, you could specify 1:4 for the fourth parameter of chm.

Note the zero in the data string, to hide the lines for the series: `chd=t0`. This indicates that all chart data is to be used for markers.

The *<which_point>* parameter is blank, which draws candlesticks on all data points.

```
chd=t0:
  0, 5,10, 7,12, 6|
 35,25,45,47,24,46|
 15,40,30,27,39,54|
 70,55,63,59,80,60
chm=F,0000FF,0,,20
```

Here's an example of the same chart, using the default colors, and removing the first and last item.

This is a compound chart: it is a combination of a line chart (the base chart type) and candlestick markers. The value of 1 in `cht=t:1` means that all data series after the first should be hidden from the basic chart type (line chart). The first 1 in `chm=F,,1,1:4,20` means that the candlestick data comes from series 2, 3, 4, and 5 (the 1 is zero-based). See *Compound Charts* (https://developers.google.com/chart/image/docs/gallery/compound_charts?hl=es) for more information on how to draw a compound chart like this.
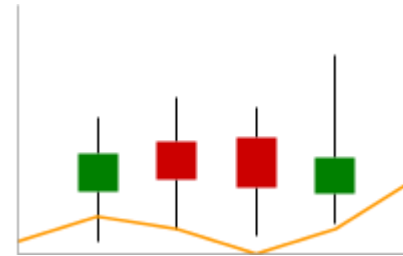
```
cht=lc
chm=
  F,,1,1:4,20
chd=t1:
  15,40,30,27,39,54|
    ...
```

Back to top (#top)

Line Markers `chm=D` [*Bar, Candlestick, Line, Radar, Scatter*]

You can add a line that traces data in your chart. Most often, this is used in compound charts (https://developers.google.com/chart/image/docs/gallery/compound_charts?hl=es).

To add multiple lines (or combine this with any other `chm` markers), separate the chm parameter sets using a pipe (|) delimiter. You cannot make a dashed line marker with this parameter.

Syntax

```
chm=
  D,<color>,<series_index>,<which_points>,<width>,<opt_z_order>
```

D

>   Indicates that this is a line marker.

<color>

>   The color of the line, in RRGGBB hexadecimal format (https://developers.google.com/chart/image/docs/chart_params?hl=es#gcharts_rgb).

<series_index>

>   The index of the data series used to draw the line. The data series index is `0` for the first data series, `1` for the second data series, and so on.

<which_points>

>   Which points in a series to use to draw the line. Use one of the following values:
>   - `0` - Use all the points in the series.
>   - *`start:end`* - Use a specific range of points in the series, from the *start* to *end*, inclusive (zero-based index). You can also use floating point values to specify intermediate points, or leave *start* or *end* blank to indicate the first or last data point, respectively. *start* and *end* can be negative, as a reverse index from the last value. If both *start* and *end* are negative, be sure to write them in increasing value (for example, -6:-1).

<size>

The width of the line in pixels.
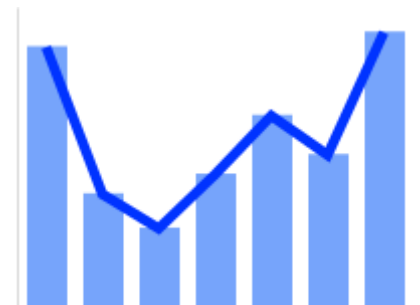
<opt_z_order>

[*Optional*] The layer on which to draw the marker, compared to other markers and all other chart elements. This is a floating point number from -1.0 to 1.0, inclusive, where -1.0 is the bottom and 1.0 is the top. Chart elements (lines and bars) are just lower than zero. If two markers have the same value, they are drawn in the order given by the URL. *Default value is 0.0* (just above the chart elements).

Examples

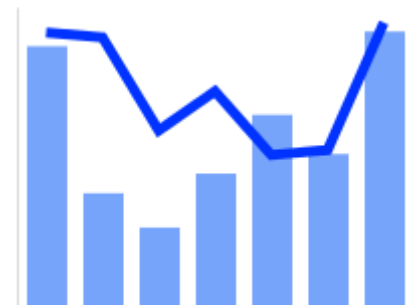| Description	Example
This is an example of drawing a marker line on a bar chart. The z-order is set to **1**, so the line is drawn on top of the bars. This example uses the same data for both the bars and the data line.	`chm=D,0033FF,0,0,5,1` `chd=s:1XQbnf4`
This is the same bar chart, but with an additional data series just for the line. This is an example of a compound chart. Compound charts are drawn by adding additional data series to the **chd** parameter, plus a value to **chd** telling the chart to "ignore" the additional data series. See *Compound Charts* (https://developers.google.com/chart/image/docs/gallery/compound_charts?hl=es) for more information.	`chm=D,0033FF,1,0,5,1` `chd=s1:1XQbnf4,43ksfg6`

Back to top (#top)

Data Functions `chfd` [*All* `chd` *charts*]

You can specify a custom function to run over chart data using muParser function syntax (http://muparser.sourceforge.net/mup_features.html#idDef3). The data used in the function can come from one of two sources:

- A data series from `chd` - The data is pulled from the specified series in `chd`.
- A range of values declared in the `chfd` parameter itself - You declare a start, stop, and step value for an arbitrary data range.

It is important to note that in all cases you must assign the output to an existing series in `chd`; that series will be overwritten by the function output. The chart is rendered only after all functions have been processes, so if you assign multiple functions to output to the same data series, the functions will be run in the order given, but only the output of the final function will be plotted on the chart. Note that you can chain functions, so that one function can take as input a series that was output by a previous function.

To assign colors or `chm` markers to a function line, assign the colors or markers to the function's series index. Note that markers are placed according to the data after it has been manipulated by the function.

Syntax

```
chfd=
    <output_series_index>,<function_data>,<function_string>
    |...|
    <output_series_index>,<function_data>,<function_string>
```

<output_series_index>

Zero-based index of a data series in `chd` to which the function output will be written. Any existing data will be completely overwritten by the function output. If a series is not being used as input for a function, best practice is to assign a single dummy value to that

series.

<function_data>

The variables and data to plot. The data can be from a range that you define, or from one of the `chd` data series. You can specify multiple variables for each function, using a ; (semicolon) delimiter for multiple variables. Note that if you define multiple variables for a single function, and these variables have a different number of points, the function will stop when it reaches the first endpoint. For example, if a function defines both x=1—5 step 1 and y=1—10 step 1, the function will end when it reaches the fifth point.

> `<variable_name>,<input_series_index>`
>
> > **OR**
>
> `<variable_name>,<start>,<end>,<step>`

- *variable_name* - An arbitrary string name for the variable. Use this in the function defined by *function_string*.

- *input_series_index* - The index of a `chd` data series to use as input data.

- *start* - The numeric start value of a range.

- *end* - The numeric end value of a range.

- *step* - The numeric step value from *start* to *end*. Can be positive or negative, but cannot be zero.

Examples: `x,0,100,1` declares a variable named `x` with values 0, 1, 2, ... 100. `x,0,100,1;r,0,3.1,.1` declares the same `x` variable plus a variable named `r` with values 0, 0.1, 0.2, ..., 3.0, 3.1. `x,0` declares a variable named x that uses the data from the first `chd` series. These variables will be used by *function_string*. They will not be plotted on the graph unless you specify them in *function_string*. The smaller the step, the smoother your graph.

<function_string>

Your function, written in the [muParser syntax](http://muparser.sourceforge.net/mup_features.html#idDef3) (http://muparser.sourceforge.net/mup_features.html#idDef3). The function is applied to the variables and data specified in *variable_data*. You can only reference the variables declared in this local function set, not in another piped set of `chfd` parameters. Summary muParser functions are not supported (min, max sum, avg). **IMPORTANT:** Remember to use `%2B` instead of `+` in your functions!

Examples

Description

A simple sine wave. Some things to notice:

- `chd=t:-1` - We use a dummy variable for the chart data, because our data is declared in the `chfd` parameter.

- `chco=FF0000` - Red is specified for the first series. Even though we don't use the data from `chd`, the corresponding color for that series will be used for tha

- `chfd=0,x,0,11,0.1,sin(x)*50%2B50` - We declare one variable, called x, with values 0—11, incremented by 0.1. It is assigned to the first series, which

This line uses data from the `chd` parameter.

A mix of function and non-function lines.

Notice how the colors are specified by the series color parameter `chco`.

Notice the placement of markers on the function output; the data points are calculated from *start*, *end*, and *step*, so if your range is 0—11 step 0.1, point 0 is 0,
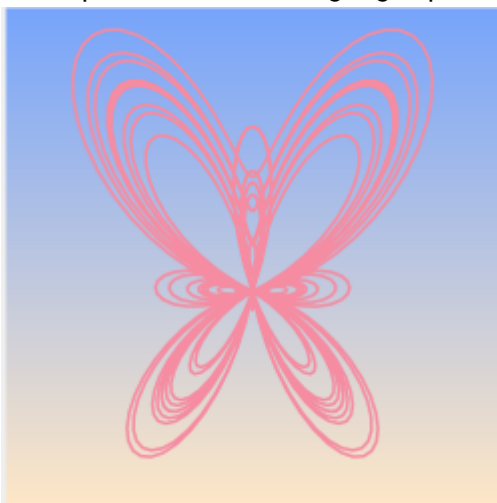
To define a function in two dimensions, use an `lxy` chart, assign two dummy series, and assign a function to each.

- `0,x,0,10,0.1,sin(x)*50%2B50` - Series 0 (the x-axis values) has a variable named x, with values from 0—10, step 0.1, and a function sin(x)*50 + 50
- `0,y,0,10,0.1,sin(y)*50%2B50` - Series 0 (the y-axis values) has a variable named y, with values from 0—10, step 0.1, and a function sin(x)*50 + 50

The `chfd` parameter can really let you express your creativity.
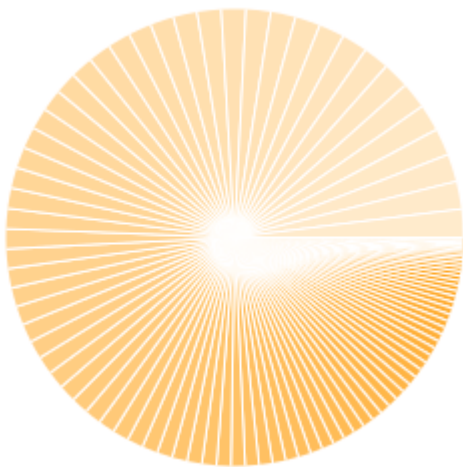
(https://developers.google.com/chart/image/docs/chart_playground?url=http%3A%2F%2Fchart.googleapis.com%2F
(https://developers.google.com/chart/image/docs/chart_playground?
url=http%3A%2F%2Fchart.googleapis.com%2Fchart%3Fcht%3Dr%26chco%3D000000%26chd%3Dt%3A0%7C0%7C0%7C0%7C0%7C0%7C0%7C0%7C0%26chs%

(https://developers.google.com/chart/image/docs/chart_playground?url=http%3A%2F%2Fchart.googleapis.com%2F

(https://developers.google.com/chart/image/docs/chart_playground?url=http%3A%2F%2Fchart.googleapis.com%2Fchart%3Fcht%3Dlxy%26chs%3D450x450

(https://developers.google.com/chart/image/docs/chart_playground?url=http%3A%2F%2Fchart.googleapis.com%2Fchart%3Fcht%3Dlxy%26chs%3D450x450
(https://developers.google.com/chart/image/docs/chart_playground?url=http%3A%2F%2Fchart.googleapis.com%2Fchart%3Fcht%3Dlxy%26chs%3D450x450

Try clicking these images to open and play with them in the chart playground; you'll get hooked!

Back to top (#top)

Última actualización: Febrero 23, 2017.