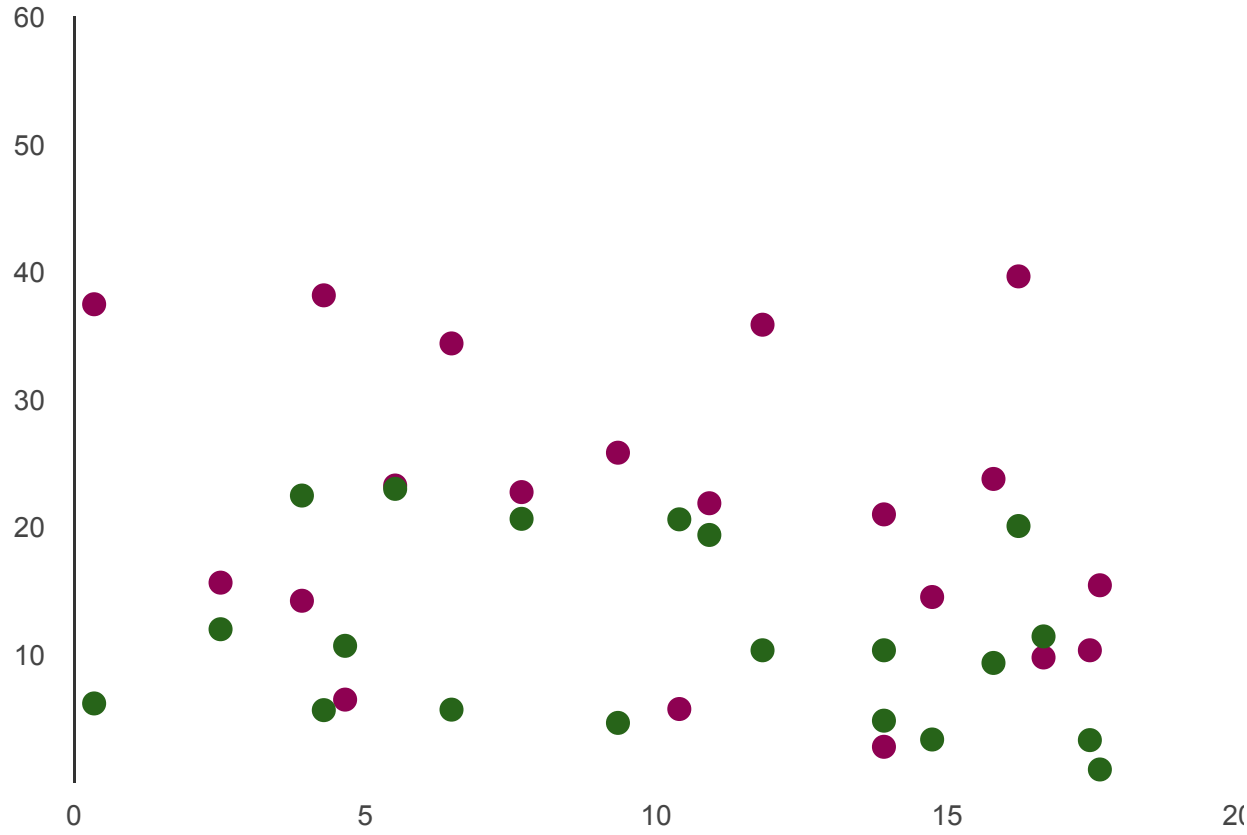
Animation

This page describes how to animate modifications made to a chart, instead of applying them instantly.



Contents

Overview

Google charts can animate smoothly in one of two ways, either on startup when you first draw the chart, or when you redraw a chart after making a change in data or options.

To animate on startup:

1. Set your chart data and options. Be sure to set an animation duration and easing type.

2. Set `animation: {"startup": true}` — setting this in your options will cause your chart to start with series values drawn at the baseline, and animate out to their final state.
3. Call `chart.draw()`, passing in your data and options.

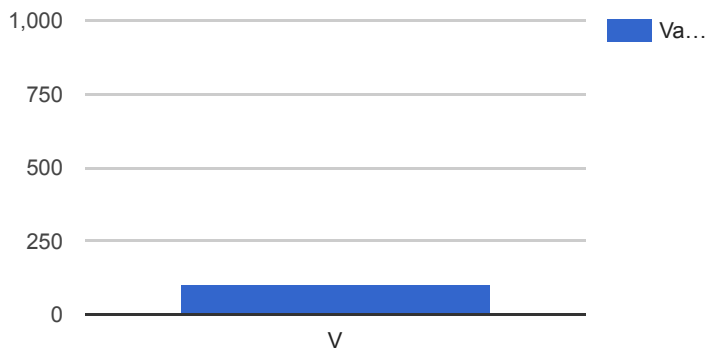
To animate a transition:

1. Start with an already rendered chart.
2. Modify the data table or options. Different chart types support different modifications; see Supported Modifications (#Supported_Modifications) to learn what modifications are supported for what chart types.
3. Specify the transition behavior using the animation (#Transition_Behavior) option.
4. Call `chart.draw()` on your chart to transition to the new values.

Here is a simple example which changes the single value presented in a bar chart upon each click on a button:

```
function init() {
 var options = {
   width: 400,
   height: 240,
   animation:{
     duration: 1000,
     easing: 'out',
   },
   vAxis: {minValue:0, maxValue:1000}
 };
 var data = new google.visualization.DataTable();
 data.addColumn('string', 'N');
 data.addColumn('number', 'Value');
 data.addRow(['V', 200]);

 var chart = new google.visualization.ColumnChart(
     document.getElementById('visualization'));
 var button = document.getElementById('b1');

 function drawChart() {
   // Disabling the button while the chart is drawing.
   button.disabled = true;
   google.visualization.events.addListener(chart, 'ready',
       function() {
         button.disabled = false;
       });
   chart.draw(data, options);
 }
```

```
  button.onclick = function() {
    var newValue = 1000 - data.getValue(0, 1);
    data.setValue(0, 1, newValue);
    drawChart();
  }
  drawChart();
}
```



CHANGE VALUE

Supported Modifications

Support for different types of transitions differs from one chart to another. The different types of transitions are:

- Changes to data table values only. The number of rows and columns is the same, and the columns maintain their original types and roles.
- Addition or removal of columns (series).
- Addition or removal of rows (categories).
- Changes to chart options. Currently, options that will animate on change include:
  - the view window (`vAxis.viewWindow.min`, `vAxis.viewWindow.max`, `hAxis.viewWindow.min`, `hAxis.viewWindow.max`) — Changing the view window is useful for achieving "zoom" and continuous "drift" effects (see examples below)
  - `vAxis.ticks` and `hAxis.ticks` values
  - `vAxis.gridlines.count` and `hAxis.gridlines.count`

- `vAxis.direction` and `hAxis.direction`
- `vAxis.baseline` and `hAxis.baseline`
- `vAixs.logScale` and `hAxis.logScale`
- chart size (`height` and `width`)
- chart area (`chartArea.height`, `chartArea.width`, `chartArea.top`, `chartArea.left`)

Modification Type	Valid Chart Types
Value changes	ScatterChart, LineChart, AreaChart, BarChart, BubbleChart, ColumnChart, CandlestickChart, SteppedAreaChart, ComboChart, Gauge
Adding/removing rows	ScatterChart, LineChart, AreaChart, BubbleChart, ComboChart (with line/area series only)
Adding/removing columns	ScatterChart, LineChart, AreaChart, BarChart, BubbleChart, ColumnChart, CandlestickChart, SteppedAreaChart, ComboChart
Modifying chart options	ScatterChart, LineChart, AreaChart, BarChart, BubbleChart, ColumnChart, CandlestickChart, SteppedAreaChart, ComboChart

Transition behavior

Name	
animation.duration	The duration of the animation, in milliseconds. For details, see the animation documentation (https://google-developers.appspot.com/chart/interactive/docs/animation) . **Type:** number **Default:** 0
animation.easing	The easing function applied to the animation. The following options are available: • 'linear' - Constant speed. • 'in' - Ease in - Start slow and speed up. • 'out' - Ease out - Start fast and slow down. • 'inAndOut' - Ease in and out - Start slow, speed up, then slow down.

	Type: string **Default:** 'linear'
animation.startup	Determines if the chart will animate on the initial draw. If **`true`**, the chart will start at the baseline and animate to its final state. **Type:** boolean **Default** false

Events

When drawing a chart, a 'ready' event is fired once the chart is ready for external method calls. The chart will fire the **`animationfinish`** event when the transition is complete.

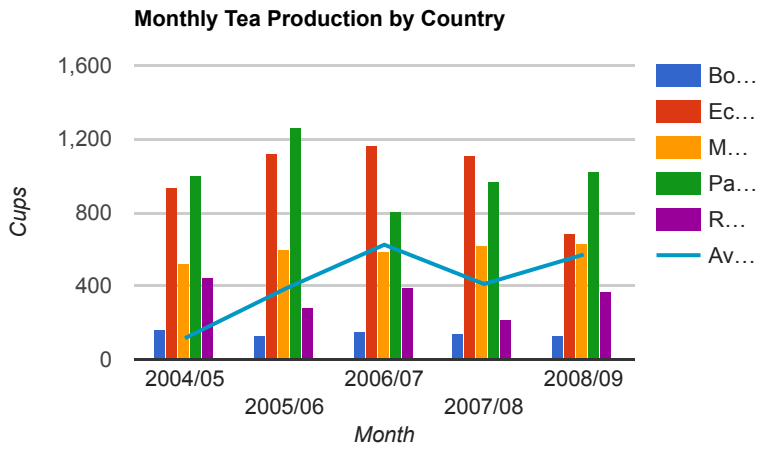
Name	
`animationfinish`	Fired when transition animation is complete. **Properties:** none

Examples

Value changes

```
// Some raw data (not necessarily accurate)
 var rowData1 = [['Month', 'Bolivia', 'Ecuador', 'Madagascar', 'Papua  Gui
                  'Rwanda', 'Average'],
                 ['2004/05', 165, 938, 522, 998, 450, 114.6],
                 ['2005/06', 135, 1120, 599, 1268, 288, 382],
                 ['2006/07', 157, 1167, 587, 807, 397, 623],
                 ['2007/08', 139, 1110, 615, 968, 215, 409.4],
                 ['2008/09', 136, 691, 629, 1026, 366, 569.6]];
 var rowData2 = [['Month', 'Bolivia', 'Ecuador', 'Madagascar', 'Papua  Gui
                  'Rwanda', 'Average'],
                 ['2004/05', 122, 638, 722, 998, 450, 614.6],
                 ['2005/06', 100, 1120, 899, 1268, 288, 682],
                 ['2006/07', 183, 167, 487, 207, 397, 623],
                 ['2007/08', 200, 510, 315, 1068, 215, 609.4],
                 ['2008/09', 123, 491, 829, 826, 366, 569.6]];

 // Create and populate the data tables.
 var data = [];
 data[0] = google.visualization.arrayToDataTable(rowData1);
```

```
  data[1] = google.visualization.arrayToDataTable(rowData2);

  var options = {
    width: 400,
    height: 240,
    vAxis: {title: "Cups"},
    hAxis: {title: "Month"},
    seriesType: "bars",
    series: {5: {type: "line"}},
    animation:{
      duration: 1000,
      easing: 'out'
    },
  };
  var current = 0;
  // Create and draw the visualization.
  var chart = new google.visualization.ComboChart(document.getElementById('v
  var button = document.getElementById('b1');
  function drawChart() {
    // Disabling the button while the chart is drawing.
    button.disabled = true;
    google.visualization.events.addListener(chart, 'ready',
        function() {
          button.disabled = false;
          button.value = 'Switch to ' + (current ? 'Tea' : 'Coffee');
        });
    options['title'] = 'Monthly ' + (current ? 'Coffee' : 'Tea') + ' Produc

    chart.draw(data[current], options);
  }
  drawChart();

  button.onclick = function() {
    current = 1 - current;
    drawChart();
  }
```

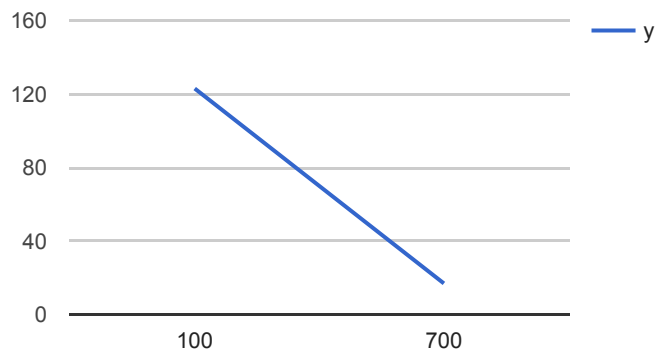


SWITCH TO TEA

Adding and removing rows

```
var options = {
  width: 400,
  height: 240,
  vAxis: {minValue:0, maxValue:100},
  animation: {
    duration: 1000,
    easing: 'in'
  }
};

var chart = new google.visualization.LineChart(
    document.getElementById('visualization'));
var data = new google.visualization.DataTable();
data.addColumn('string', 'x');
data.addColumn('number', 'y');
data.addRow(['100', 123]);
data.addRow(['700', 17]);
var button = document.getElementById('b1');
function drawChart() {
  // Disabling the button while the chart is drawing.
  button.disabled = true;
  google.visualization.events.addListener(chart, 'ready',
      function() {
        button.disabled = false;
      });
  chart.draw(data, options);
}

button.onclick = function() {
```

```
    if (data.getNumberOfRows() > 5) {
      data.removeRow(Math.floor(Math.random() * data.getNumberOfRows()));
    }
    // Generating a random x, y pair and inserting it so rows are sorted.
    var x = Math.floor(Math.random() * 1000);
    var y = Math.floor(Math.random() * 100);
    var where = 0;
    while (where < data.getNumberOfRows() && parseInt(data.getValue(where,
      where++;
    }
    data.insertRows(where, [[x.toString(), y]]);
    drawChart();
  }
  drawChart();
```

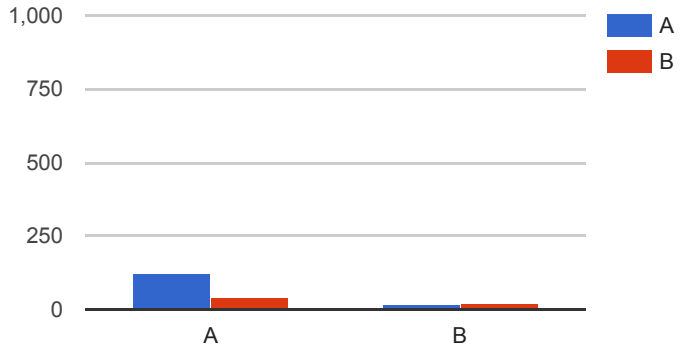


ADD AND/OR REMOVE ROWS

Adding and removing columns

```
  var options = {
    width: 400,
    height: 240,
    vAxis: {minValue:0, maxValue:1000},
    animation: {
      duration: 1000,
      easing: 'out'
    }
  };

  var chart = new google.visualization.ColumnChart(
      document.getElementById('visualization'));
  var chars = 'ABCDEFGHIJKLMNOPQRSTUVWXTZabcdefghiklmnopqrstuvwxyz';
```

```
var data = new google.visualization.DataTable();
data.addColumn('string', 'x');
data.addColumn('number', 'A');
data.addColumn('number', 'B');
data.addRow(['A', 123, 40]);
data.addRow(['B', 17, 20]);
var addButton = document.getElementById('b1');
var removeButton = document.getElementById('b2');
function drawChart() {
  // Disabling the buttons while the chart is drawing.
  addButton.disabled = true;
  removeButton.disabled = true;
  google.visualization.events.addListener(chart, 'ready',
      function() {
        // Enabling only relevant buttons.
        addButton.disabled = (data.getNumberOfColumns() - 1) >= chars.leng
        removeButton.disabled = (data.getNumberOfColumns() - 1) < 2;
      });
  chart.draw(data, options);
}
function shuffleAndDrawChart() {
  for (var i = 0; i < data.getNumberOfRows(); ++i) {
    for (var j = 1; j < data.getNumberOfColumns(); ++j) {
      var num = Math.floor(Math.random() * 1000);
      data.setValue(i, j, num);
    }
  }
  drawChart();
}
addButton.onclick = function() {
  data.addColumn('number', chars[data.getNumberOfColumns() - 1]);
  shuffleAndDrawChart();
}
removeButton.onclick = function() {
  data.removeColumn(data.getNumberOfColumns() - 1);
  shuffleAndDrawChart();
}
drawChart();
```



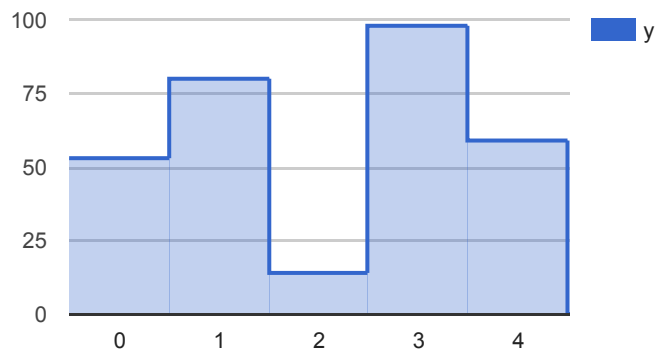
ADD COLUMN AND SHUFFLE VALUES

REMOVE COLUMN AND SHUFFLE VALUES

Changing the view window

```
var options = {
  width: 400,
  height: 240,
  animation: {
    duration: 1000,
    easing: 'in'
  },
  hAxis: {viewWindow: {min:0, max:5}}
};

var chart = new google.visualization.SteppedAreaChart(
    document.getElementById('visualization'));
var data = new google.visualization.DataTable();
data.addColumn('string', 'x');
data.addColumn('number', 'y');
var MAX = 10;
for (var i = 0; i < MAX; ++i) {
  data.addRow([i.toString(), Math.floor(Math.random() * 100)]);
}
var prevButton = document.getElementById('b1');
var nextButton = document.getElementById('b2');
var changeZoomButton = document.getElementById('b3');
function drawChart() {
  // Disabling the button while the chart is drawing.
  prevButton.disabled = true;
  nextButton.disabled = true;
  changeZoomButton.disabled = true;
  google.visualization.events.addListener(chart, 'ready',
      function() {
```

```
        prevButton.disabled = options.hAxis.viewWindow.min <= 0;
        nextButton.disabled = options.hAxis.viewWindow.max >= MAX;
        changeZoomButton.disabled = false;
      });
  chart.draw(data, options);
}

prevButton.onclick = function() {
  options.hAxis.viewWindow.min -= 1;
  options.hAxis.viewWindow.max -= 1;
  drawChart();
}
nextButton.onclick = function() {
  options.hAxis.viewWindow.min += 1;
  options.hAxis.viewWindow.max += 1;
  drawChart();
}
var zoomed = false;
changeZoomButton.onclick = function() {
  if (zoomed) {
    options.hAxis.viewWindow.min = 0;
    options.hAxis.viewWindow.max = 5;
  } else {
    options.hAxis.viewWindow.min = 0;
    options.hAxis.viewWindow.max = MAX;
  }
  zoomed = !zoomed;
  drawChart();
}
drawChart();
```



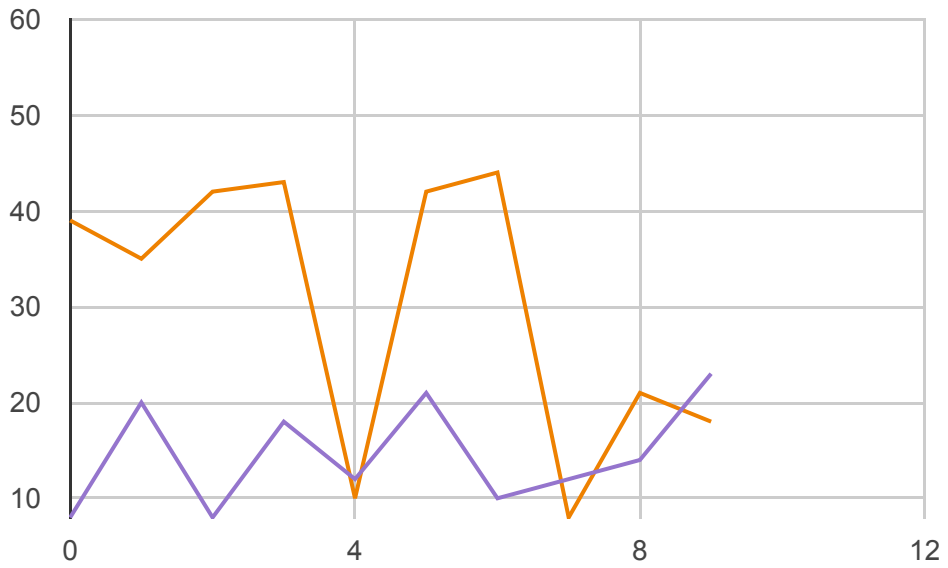
PREVIOUS NEXT CHANGE ZOOM

Changing the chart size

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

  function drawChart () {

    function genData () {
      var array = [];

      for (var i = 0; i < 10; i++) {
        var x = i,
            y = Math.floor(Math.random() * 50),
            z = Math.floor(Math.random() * 25);
        array.push([x, y, z]);
      }
      return array;
    }

    function doubleIt() {
      options.chartArea.height = '100%';
      options.chartArea.width = '100%';
    }

    function halveIt() {
      options.chartArea.height = '50%';
      options.chartArea.width = '50%';
    }

    function goTo50() {
      options.chartArea.left = '50%';
      options.chartArea.top = '50%';
    }

    function goTo10() {
      options.chartArea.left = '10%';
      options.chartArea.top = '10%';
    }

    var data = new google.visualization.DataTable();
    data.addColumn('number', 'X');
    data.addColumn('number', 'Y');
    data.addColumn('number', 'Z');

    data.addRows(genData());

    var options = {
      height: 500,
```

```
      chartArea: {
        height: '50%',
        width: '50%',
        top: '10%',
        left: '10%'
      },
      colors: ['#ee8100', '#9575cd'],
      animation: {
        duration: 1500,
        easing: 'linear',
        startup: true
      },
      vAxis: {
        ticks: [10, 20, 30, 40, 50, 60],
        gridlines: {color: '#ccc'}
      },
      hAxis: {
        ticks: [0, 4, 8, 12],
        gridlines: {color: '#ccc'}
      },
      legend: {position: 'none'},
      enableInteractivity: false
    };

    var chart = new google.visualization.LineChart(document.getElementById(


    chart.draw(data, options);

    var btns = document.querySelector('#btns');
    btns.onclick = function (e) {
      switch(e.target.id) {
          case "size":
            options.chartArea.height === '50%' ? doubleIt() : halveIt();
            break;
          case "slide":
            options.chartArea.left === '10%' ? goTo50() : goTo10();
      }

      chart.draw(data, options);
    }

  }
```

CHANGE THE CHART SIZE SLIDE THE CHART



上次更新日期：二月 23, 2017