* [Chart.js](http://docs.google.com/)
* [Getting Started](http://docs.google.com/getting-started/)
  + [Installation](http://docs.google.com/getting-started/installation.html)
  + [Integration](http://docs.google.com/getting-started/integration.html)
  + [Usage](http://docs.google.com/getting-started/usage.html)
* [General](http://docs.google.com/general/)
  + [Responsive](http://docs.google.com/general/responsive.html)
  + [Pixel Ratio](http://docs.google.com/general/device-pixel-ratio.html)
  + [Interactions](http://docs.google.com/general/interactions/)
    - [Events](http://docs.google.com/general/interactions/events.html)
    - [Modes](http://docs.google.com/general/interactions/modes.html)
  + [Options](http://docs.google.com/general/options.html)
  + [Colors](http://docs.google.com/general/colors.html)
  + [Fonts](http://docs.google.com/general/fonts.html)
* [Configuration](http://docs.google.com/configuration/)
  + [Animations](http://docs.google.com/configuration/animations.html)
  + [Layout](http://docs.google.com/configuration/layout.html)
  + [Legend](http://docs.google.com/configuration/legend.html)
  + [Title](http://docs.google.com/configuration/title.html)
  + [Tooltip](http://docs.google.com/configuration/tooltip.html)
  + [Elements](http://docs.google.com/configuration/elements.html)
* [Charts](http://docs.google.com/charts/)
  + [Line](http://docs.google.com/charts/line.html)
  + [Bar](http://docs.google.com/charts/bar.html)
  + [Radar](http://docs.google.com/charts/radar.html)
  + [Doughnut & Pie](http://docs.google.com/charts/doughnut.html)
  + [Polar Area](http://docs.google.com/charts/polar.html)
  + [Bubble](http://docs.google.com/charts/bubble.html)
  + [Scatter](http://docs.google.com/charts/scatter.html)
  + [Area](http://docs.google.com/charts/area.html)
  + [Mixed](http://docs.google.com/charts/mixed.html)
* [Axes](http://docs.google.com/axes/)
  + [Cartesian](http://docs.google.com/axes/cartesian/)
    - [Category](http://docs.google.com/axes/cartesian/category.html)
    - [Linear](http://docs.google.com/axes/cartesian/linear.html)
    - [Logarithmic](http://docs.google.com/axes/cartesian/logarithmic.html)
    - [Time](http://docs.google.com/axes/cartesian/time.html)
  + [Radial](http://docs.google.com/axes/radial/)
    - [Linear](http://docs.google.com/axes/radial/linear.html)
  + [Labelling](http://docs.google.com/axes/labelling.html)
  + [Styling](http://docs.google.com/axes/styling.html)
* [Developers](http://docs.google.com/)
  + [Chart.js API](http://docs.google.com/api.html)
  + [Updating Charts](http://docs.google.com/updates.html)
  + [Plugins](http://docs.google.com/plugins.html)
  + [New Charts](http://docs.google.com/charts.html)
  + [New Axes](http://docs.google.com/axes.html)
  + [Contributing](http://docs.google.com/contributing.html)
* [Additional Notes](http://docs.google.com/notes/)
  + [Comparison Table](http://docs.google.com/notes/comparison.html)
  + [Popular Extensions](http://docs.google.com/notes/extensions.html)
  + [License](http://docs.google.com/notes/license.html)
* [Published with GitBook](https://www.gitbook.com)

[**Updating Charts**](http://docs.google.com/)

Updating Charts

It's pretty common to want to update charts after they've been created. When the chart data or options are changed, Chart.js will animate to the new data values and options.

## Adding or Removing Data

Adding and removing data is supported by changing the data array. To add data, just add data into the data array as seen in this example.

function addData(chart, label, data) {  
 chart.data.labels.push(label);  
 chart.data.datasets.forEach((dataset) => {  
 dataset.data.push(data);  
 });  
 chart.update();  
}  
  
function removeData(chart) {  
 chart.data.labels.pop();  
 chart.data.datasets.forEach((dataset) => {  
 dataset.data.pop();  
 });  
 chart.update();  
}

## Updating Options

To update the options, mutating the options property in place or passing in a new options object are supported.

* If the options are mutated in place, other option properties would be preserved, including those calculated by Chart.js.
* If created as a new object, it would be like creating a new chart with the options - old options would be discarded.

function updateConfigByMutating(chart) {  
 chart.options.title.text = 'new title';  
 chart.update();  
}  
  
function updateConfigAsNewObject(chart) {  
 chart.options = {  
 responsive: true,  
 title:{  
 display:true,  
 text: 'Chart.js'  
 },  
 scales: {  
 xAxes: [{  
 display: true  
 }],  
 yAxes: [{  
 display: true  
 }]  
 }  
 }  
 chart.update();  
}

Scales can be updated separately without changing other options. To update the scales, pass in an object containing all the customization including those unchanged ones.

Variables referencing any one from chart.scales would be lost after updating scales with a new id or the changed type.

function updateScales(chart) {  
 var xScale = chart.scales['x-axis-0'];  
 var yScale = chart.scales['y-axis-0'];  
 chart.options.scales = {  
 xAxes: [{  
 id: 'newId',  
 display: true  
 }],  
 yAxes: [{  
 display: true,  
 type: 'logarithmic'  
 }]  
 }  
 chart.update();  
 // need to update the reference  
 xScale = chart.scales['newId'];  
 yScale = chart.scales['y-axis-0'];  
}

You can also update a specific scale either by specifying its index or id.

function updateScale(chart) {  
 chart.options.scales.yAxes[0] = {  
 type: 'logarithmic'  
 }  
 chart.update();  
}

Code sample for updating options can be found in [toggle-scale-type.html](http://docs.google.com/samples/scales/toggle-scale-type.html).

## Preventing Animations

Sometimes when a chart updates, you may not want an animation. To achieve this you can call update with a duration of 0. This will render the chart synchronously and without an animation.

results matching ""

No results matching ""