 [Chart.js](http://docs.google.com/docs/3.9.1/)

[Home](http://docs.google.com/docs/3.9.1/)

[API](http://docs.google.com/docs/3.9.1/api/)

[Samples](http://docs.google.com/docs/3.9.1/samples/)

Ecosystem Ecosystem

* [Awesome (opens new window)](https://github.com/chartjs/awesome)
* [Slack (opens new window)](https://chartjs-slack.herokuapp.com/)
* [Stack Overflow (opens new window)](https://stackoverflow.com/questions/tagged/chart.js)

[GitHub (opens new window)](https://github.com/chartjs/Chart.js)

[Home](http://docs.google.com/docs/3.9.1/)

[API](http://docs.google.com/docs/3.9.1/api/)

[Samples](http://docs.google.com/docs/3.9.1/samples/)

Ecosystem Ecosystem

* [Awesome (opens new window)](https://github.com/chartjs/awesome)
* [Slack (opens new window)](https://chartjs-slack.herokuapp.com/)
* [Stack Overflow (opens new window)](https://stackoverflow.com/questions/tagged/chart.js)

[GitHub (opens new window)](https://github.com/chartjs/Chart.js)

* [Information](http://docs.google.com/docs/3.9.1/samples/information.html)
* Bar Charts
* Line Charts
* Other charts
* Area charts
* Scales
* Scale Options
* Legend
* Title
* Subtitle
* Tooltip
* Scriptable Options
* Animations
  + [Delay](http://docs.google.com/docs/3.9.1/samples/animations/delay.html)
  + [Drop](http://docs.google.com/docs/3.9.1/samples/animations/drop.html)
  + [Loop](http://docs.google.com/docs/3.9.1/samples/animations/loop.html)
  + [Progressive Line](http://docs.google.com/docs/3.9.1/samples/animations/progressive-line.html)
  + [Progressive Line With Easing](http://docs.google.com/docs/3.9.1/samples/animations/progressive-line-easing.html)
* Advanced
* Plugins
* [Utils](http://docs.google.com/docs/3.9.1/samples/utils.html)

[**#**](#gjdgxs) Drop

config setup actions

const config = { type: 'line', data: data, options: { animations: { y: { easing: 'easeInOutElastic', from: (ctx) => { if (ctx.type === 'data') { if (ctx.mode === 'default' && !ctx.dropped) { ctx.dropped = true; return 0; } } } } }, }, };

const config = {  
 type: 'line',  
 data: data,  
 options: {  
 animations: {  
 y: {  
 easing: 'easeInOutElastic',  
 from: (ctx) => {  
 if (ctx.type === 'data') {  
 if (ctx.mode === 'default' && !ctx.dropped) {  
 ctx.dropped = true;  
 return 0;  
 }  
 }  
 }  
 }  
 },  
 },  
};

const DATA\_COUNT = 7; const NUMBER\_CFG = {count: DATA\_COUNT, min: -100, max: 100}; const labels = Utils.months({count: 7}); const data = { labels: labels, datasets: [ { label: 'Dataset 1', animations: { y: { duration: 2000, delay: 500 } }, data: Utils.numbers(NUMBER\_CFG), borderColor: Utils.CHART\_COLORS.red, backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.red, 0.5), fill: 1, tension: 0.5 }, { label: 'Dataset 2', data: Utils.numbers(NUMBER\_CFG), borderColor: Utils.CHART\_COLORS.blue, backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.blue, 0.5), } ] };

const DATA\_COUNT = 7;  
const NUMBER\_CFG = {count: DATA\_COUNT, min: -100, max: 100};  
const labels = Utils.months({count: 7});  
const data = {  
 labels: labels,  
 datasets: [  
 {  
 label: 'Dataset 1',  
 animations: {  
 y: {  
 duration: 2000,  
 delay: 500  
 }  
 },  
 data: Utils.numbers(NUMBER\_CFG),  
 borderColor: Utils.CHART\_COLORS.red,  
 backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.red, 0.5),  
 fill: 1,  
 tension: 0.5  
 },  
 {  
 label: 'Dataset 2',  
 data: Utils.numbers(NUMBER\_CFG),  
 borderColor: Utils.CHART\_COLORS.blue,  
 backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.blue, 0.5),  
 }  
 ]  
};

const actions = [ { name: 'Randomize', handler(chart) { chart.data.datasets.forEach(dataset => { dataset.data = Utils.numbers({count: chart.data.labels.length, min: -100, max: 100}); }); chart.update(); } }, { name: 'Add Dataset', handler(chart) { const data = chart.data; const dsColor = Utils.namedColor(chart.data.datasets.length); const newDataset = { label: 'Dataset ' + (data.datasets.length + 1), backgroundColor: Utils.transparentize(dsColor, 0.5), borderColor: dsColor, data: Utils.numbers({count: data.labels.length, min: -100, max: 100}), }; chart.data.datasets.push(newDataset); chart.update(); } }, { name: 'Add Data', handler(chart) { const data = chart.data; if (data.datasets.length > 0) { data.labels = Utils.months({count: data.labels.length + 1}); for (let index = 0; index < data.datasets.length; ++index) { data.datasets[index].data.push(Utils.rand(-100, 100)); } chart.update(); } } }, { name: 'Remove Dataset', handler(chart) { chart.data.datasets.pop(); chart.update(); } }, { name: 'Remove Data', handler(chart) { chart.data.labels.splice(-1, 1); // remove the label first chart.data.datasets.forEach(dataset => { dataset.data.pop(); }); chart.update(); } } ];

const actions = [  
 {  
 name: 'Randomize',  
 handler(chart) {  
 chart.data.datasets.forEach(dataset => {  
 dataset.data = Utils.numbers({count: chart.data.labels.length, min: -100, max: 100});  
 });  
 chart.update();  
 }  
 },  
 {  
 name: 'Add Dataset',  
 handler(chart) {  
 const data = chart.data;  
 const dsColor = Utils.namedColor(chart.data.datasets.length);  
 const newDataset = {  
 label: 'Dataset ' + (data.datasets.length + 1),  
 backgroundColor: Utils.transparentize(dsColor, 0.5),  
 borderColor: dsColor,  
 data: Utils.numbers({count: data.labels.length, min: -100, max: 100}),  
 };  
 chart.data.datasets.push(newDataset);  
 chart.update();  
 }  
 },  
 {  
 name: 'Add Data',  
 handler(chart) {  
 const data = chart.data;  
 if (data.datasets.length > 0) {  
 data.labels = Utils.months({count: data.labels.length + 1});  
 for (let index = 0; index < data.datasets.length; ++index) {  
 data.datasets[index].data.push(Utils.rand(-100, 100));  
 }  
 chart.update();  
 }  
 }  
 },  
 {  
 name: 'Remove Dataset',  
 handler(chart) {  
 chart.data.datasets.pop();  
 chart.update();  
 }  
 },  
 {  
 name: 'Remove Data',  
 handler(chart) {  
 chart.data.labels.splice(-1, 1); // remove the label first  
 chart.data.datasets.forEach(dataset => {  
 dataset.data.pop();  
 });  
 chart.update();  
 }  
 }  
];

## [**#**](#30j0zll) Docs

* [Area](http://docs.google.com/docs/3.9.1/charts/area.html)
* [Animations](http://docs.google.com/docs/3.9.1/configuration/animations.html)
  + [animation (easing)](http://docs.google.com/docs/3.9.1/configuration/animations.html#animation)
  + [animations (from)](http://docs.google.com/docs/3.9.1/configuration/animations.html#animations-2)
* [Line](http://docs.google.com/docs/3.9.1/charts/line.html)
  + [Line Styling](http://docs.google.com/docs/3.9.1/charts/line.html#line-styling)
    - fill
    - tension
* [Options](http://docs.google.com/docs/3.9.1/general/options.html)
  + [Scriptable Options](http://docs.google.com/docs/3.9.1/general/options.html#scriptable-options)

Last Updated: 8/3/2022, 12:46:38 PM

←  [Delay](http://docs.google.com/docs/3.9.1/samples/animations/delay.html)   [Loop](http://docs.google.com/docs/3.9.1/samples/animations/loop.html)  →