 [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
* Advanced
  + [Data Decimation](http://docs.google.com/docs/3.9.1/samples/advanced/data-decimation.html)
  + [Derived Axis Type](http://docs.google.com/docs/3.9.1/samples/advanced/derived-axis-type.html)
  + [Derived Chart Type](http://docs.google.com/docs/3.9.1/samples/advanced/derived-chart-type.html)
  + [Linear Gradient](http://docs.google.com/docs/3.9.1/samples/advanced/linear-gradient.html)
  + [Programmatic Event Triggers](http://docs.google.com/docs/3.9.1/samples/advanced/programmatic-events.html)
  + [Animation Progress Bar](http://docs.google.com/docs/3.9.1/samples/advanced/progress-bar.html)
  + [Radial Gradient](http://docs.google.com/docs/3.9.1/samples/advanced/radial-gradient.html)
* Plugins
* [Utils](http://docs.google.com/docs/3.9.1/samples/utils.html)

[**#**](#gjdgxs) Animation Progress Bar

## [**#**](#30j0zll) Initial animation

## [**#**](#1fob9te) Other animations

config setup actions

const config = { type: 'line', data: data, options: { animation: { duration: 2000, onProgress: function(context) { if (context.initial) { initProgress.value = context.currentStep / context.numSteps; } else { progress.value = context.currentStep / context.numSteps; } }, onComplete: function(context) { if (context.initial) { console.log('Initial animation finished'); } else { console.log('animation finished'); } } }, interaction: { mode: 'nearest', axis: 'x', intersect: false }, plugins: { title: { display: true, text: 'Chart.js Line Chart - Animation Progress Bar' } }, }, };

const config = {  
 type: 'line',  
 data: data,  
 options: {  
 animation: {  
 duration: 2000,  
 onProgress: function(context) {  
 if (context.initial) {  
 initProgress.value = context.currentStep / context.numSteps;  
 } else {  
 progress.value = context.currentStep / context.numSteps;  
 }  
 },  
 onComplete: function(context) {  
 if (context.initial) {  
 console.log('Initial animation finished');  
 } else {  
 console.log('animation finished');  
 }  
 }  
 },  
 interaction: {  
 mode: 'nearest',  
 axis: 'x',  
 intersect: false  
 },  
 plugins: {  
 title: {  
 display: true,  
 text: 'Chart.js Line Chart - Animation Progress Bar'  
 }  
 },  
 },  
};

const initProgress = document.getElementById('initialProgress'); const progress = document.getElementById('animationProgress'); 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', data: Utils.numbers(NUMBER\_CFG), borderColor: Utils.CHART\_COLORS.red, backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.red, 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 initProgress = document.getElementById('initialProgress');  
const progress = document.getElementById('animationProgress');  
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',  
 data: Utils.numbers(NUMBER\_CFG),  
 borderColor: Utils.CHART\_COLORS.red,  
 backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.red, 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();  
 }  
 }  
];

## [**#**](#3znysh7) Docs

* [Animations](http://docs.google.com/docs/3.9.1/configuration/animations.html)
  + [Animation Callbacks](http://docs.google.com/docs/3.9.1/configuration/animations.html#animation-callbacks)
* [Data structures (labels)](http://docs.google.com/docs/3.9.1/general/data-structures.html)
* [Line](http://docs.google.com/docs/3.9.1/charts/line.html)
* [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

←  [Programmatic Event Triggers](http://docs.google.com/docs/3.9.1/samples/advanced/programmatic-events.html)   [Radial Gradient](http://docs.google.com/docs/3.9.1/samples/advanced/radial-gradient.html)  →