 [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
  + [Line Chart Boundaries](http://docs.google.com/docs/3.9.1/samples/area/line-boundaries.html)
  + [Line Chart Datasets](http://docs.google.com/docs/3.9.1/samples/area/line-datasets.html)
  + [Line Chart drawTime](http://docs.google.com/docs/3.9.1/samples/area/line-drawtime.html)
  + [Line Chart Stacked](http://docs.google.com/docs/3.9.1/samples/area/line-stacked.html)
  + [Radar Chart Stacked](http://docs.google.com/docs/3.9.1/samples/area/radar.html)
* Scales
* Scale Options
* Legend
* Title
* Subtitle
* Tooltip
* Scriptable Options
* Animations
* Advanced
* Plugins
* [Utils](http://docs.google.com/docs/3.9.1/samples/utils.html)

[**#**](#gjdgxs) Line Chart drawTime

data config setup actions

const data = { labels: generateLabels(), datasets: [ { label: 'Dataset 1', data: generateData(), borderColor: Utils.CHART\_COLORS.red, backgroundColor: Utils.CHART\_COLORS.red, fill: true }, { label: 'Dataset 2', data: generateData(), borderColor: Utils.CHART\_COLORS.blue, backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.blue), fill: true } ] };

const data = {  
 labels: generateLabels(),  
 datasets: [  
 {  
 label: 'Dataset 1',  
 data: generateData(),  
 borderColor: Utils.CHART\_COLORS.red,  
 backgroundColor: Utils.CHART\_COLORS.red,  
 fill: true  
 },  
 {  
 label: 'Dataset 2',  
 data: generateData(),  
 borderColor: Utils.CHART\_COLORS.blue,  
 backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.blue),  
 fill: true  
 }  
 ]  
};

const config = { type: 'line', data: data, options: { plugins: { filler: { propagate: false, }, title: { display: true, text: (ctx) => 'drawTime: ' + ctx.chart.options.plugins.filler.drawTime } }, pointBackgroundColor: '#fff', radius: 10, interaction: { intersect: false, } }, };

const config = {  
 type: 'line',  
 data: data,  
 options: {  
 plugins: {  
 filler: {  
 propagate: false,  
 },  
 title: {  
 display: true,  
 text: (ctx) => 'drawTime: ' + ctx.chart.options.plugins.filler.drawTime  
 }  
 },  
 pointBackgroundColor: '#fff',  
 radius: 10,  
 interaction: {  
 intersect: false,  
 }  
 },  
};

const inputs = { min: -100, max: 100, count: 8, decimals: 2, continuity: 1 }; const generateLabels = () => { return Utils.months({count: inputs.count}); }; Utils.srand(3); const generateData = () => (Utils.numbers(inputs));

const inputs = {  
 min: -100,  
 max: 100,  
 count: 8,  
 decimals: 2,  
 continuity: 1  
};  
const generateLabels = () => {  
 return Utils.months({count: inputs.count});  
};  
Utils.srand(3);  
const generateData = () => (Utils.numbers(inputs));

let smooth = false; const actions = [ { name: 'drawTime: beforeDatasetDraw (default)', handler: (chart) => { chart.options.plugins.filler.drawTime = 'beforeDatasetDraw'; chart.update(); } }, { name: 'drawTime: beforeDatasetsDraw', handler: (chart) => { chart.options.plugins.filler.drawTime = 'beforeDatasetsDraw'; chart.update(); } }, { name: 'drawTime: beforeDraw', handler: (chart) => { chart.options.plugins.filler.drawTime = 'beforeDraw'; chart.update(); } }, { name: 'Randomize', handler(chart) { chart.data.datasets.forEach(dataset => { dataset.data = generateData(); }); chart.update(); } }, { name: 'Smooth', handler(chart) { smooth = !smooth; chart.options.elements.line.tension = smooth ? 0.4 : 0; chart.update(); } } ];

let smooth = false;  
const actions = [  
 {  
 name: 'drawTime: beforeDatasetDraw (default)',  
 handler: (chart) => {  
 chart.options.plugins.filler.drawTime = 'beforeDatasetDraw';  
 chart.update();  
 }  
 },  
 {  
 name: 'drawTime: beforeDatasetsDraw',  
 handler: (chart) => {  
 chart.options.plugins.filler.drawTime = 'beforeDatasetsDraw';  
 chart.update();  
 }  
 },  
 {  
 name: 'drawTime: beforeDraw',  
 handler: (chart) => {  
 chart.options.plugins.filler.drawTime = 'beforeDraw';  
 chart.update();  
 }  
 },  
 {  
 name: 'Randomize',  
 handler(chart) {  
 chart.data.datasets.forEach(dataset => {  
 dataset.data = generateData();  
 });  
 chart.update();  
 }  
 },  
 {  
 name: 'Smooth',  
 handler(chart) {  
 smooth = !smooth;  
 chart.options.elements.line.tension = smooth ? 0.4 : 0;  
 chart.update();  
 }  
 }  
];

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

* [Area](http://docs.google.com/docs/3.9.1/charts/area.html)
  + [Configuration (drawTime)](http://docs.google.com/docs/3.9.1/charts/area.html#configuration)
* [Line](http://docs.google.com/docs/3.9.1/charts/line.html)
  + [Line Styling (tension)](http://docs.google.com/docs/3.9.1/charts/line.html#line-styling)
* [Data structures (labels)](http://docs.google.com/docs/3.9.1/general/data-structures.html)

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

←  [Line Chart Datasets](http://docs.google.com/docs/3.9.1/samples/area/line-datasets.html)   [Line Chart Stacked](http://docs.google.com/docs/3.9.1/samples/area/line-stacked.html)  →