 [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) Derived Axis Type

config setup

const config = { type: 'line', data, options: { responsive: true, scales: { x: { display: true, }, y: { display: true, type: 'log2', } } } };

const config = {  
 type: 'line',  
 data,  
 options: {  
 responsive: true,  
 scales: {  
 x: {  
 display: true,  
 },  
 y: {  
 display: true,  
 type: 'log2',  
 }  
 }  
 }  
};

const DATA\_COUNT = 12; const NUMBER\_CFG = {count: DATA\_COUNT, min: 0, max: 1000}; const labels = Utils.months({count: DATA\_COUNT}); const data = { labels: labels, datasets: [ { label: 'My First dataset', data: Utils.numbers(NUMBER\_CFG), borderColor: Utils.CHART\_COLORS.red, backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.red, 0.5), fill: false, } ], };

const DATA\_COUNT = 12;  
const NUMBER\_CFG = {count: DATA\_COUNT, min: 0, max: 1000};  
const labels = Utils.months({count: DATA\_COUNT});  
const data = {  
 labels: labels,  
 datasets: [  
 {  
 label: 'My First dataset',  
 data: Utils.numbers(NUMBER\_CFG),  
 borderColor: Utils.CHART\_COLORS.red,  
 backgroundColor: Utils.transparentize(Utils.CHART\_COLORS.red, 0.5),  
 fill: false,  
 }  
 ],  
};

## [**#**](#30j0zll) Log2 axis implementation

import {Scale, LinearScale} from 'chart.js';  
export default class Log2Axis extends Scale {  
 constructor(cfg) {  
 super(cfg);  
 this.\_startValue = undefined;  
 this.\_valueRange = 0;  
 }  
 parse(raw, index) {  
 const value = LinearScale.prototype.parse.apply(this, [raw, index]);  
 return isFinite(value) && value > 0 ? value : null;  
 }  
 determineDataLimits() {  
 const {min, max} = this.getMinMax(true);  
 this.min = isFinite(min) ? Math.max(0, min) : null;  
 this.max = isFinite(max) ? Math.max(0, max) : null;  
 }  
 buildTicks() {  
 const ticks = [];  
 let power = Math.floor(Math.log2(this.min || 1));  
 let maxPower = Math.ceil(Math.log2(this.max || 2));  
 while (power <= maxPower) {  
 ticks.push({value: Math.pow(2, power)});  
 power += 1;  
 }  
 this.min = ticks[0].value;  
 this.max = ticks[ticks.length - 1].value;  
 return ticks;  
 }  
 /\*\*  
 \* @protected  
 \*/  
 configure() {  
 const start = this.min;  
 super.configure();  
 this.\_startValue = Math.log2(start);  
 this.\_valueRange = Math.log2(this.max) - Math.log2(start);  
 }  
 getPixelForValue(value) {  
 if (value === undefined || value === 0) {  
 value = this.min;  
 }  
 return this.getPixelForDecimal(value === this.min ? 0  
 : (Math.log2(value) - this.\_startValue) / this.\_valueRange);  
 }  
 getValueForPixel(pixel) {  
 const decimal = this.getDecimalForPixel(pixel);  
 return Math.pow(2, this.\_startValue + decimal \* this.\_valueRange);  
 }  
}  
Log2Axis.id = 'log2';  
Log2Axis.defaults = {};  
// The derived axis is registered like this:  
// Chart.register(Log2Axis);

## [**#**](#1fob9te) Docs

* [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)
* [New Axes](http://docs.google.com/docs/3.9.1/developers/axes.html)

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

←  [Data Decimation](http://docs.google.com/docs/3.9.1/samples/advanced/data-decimation.html)   [Derived Chart Type](http://docs.google.com/docs/3.9.1/samples/advanced/derived-chart-type.html)  →